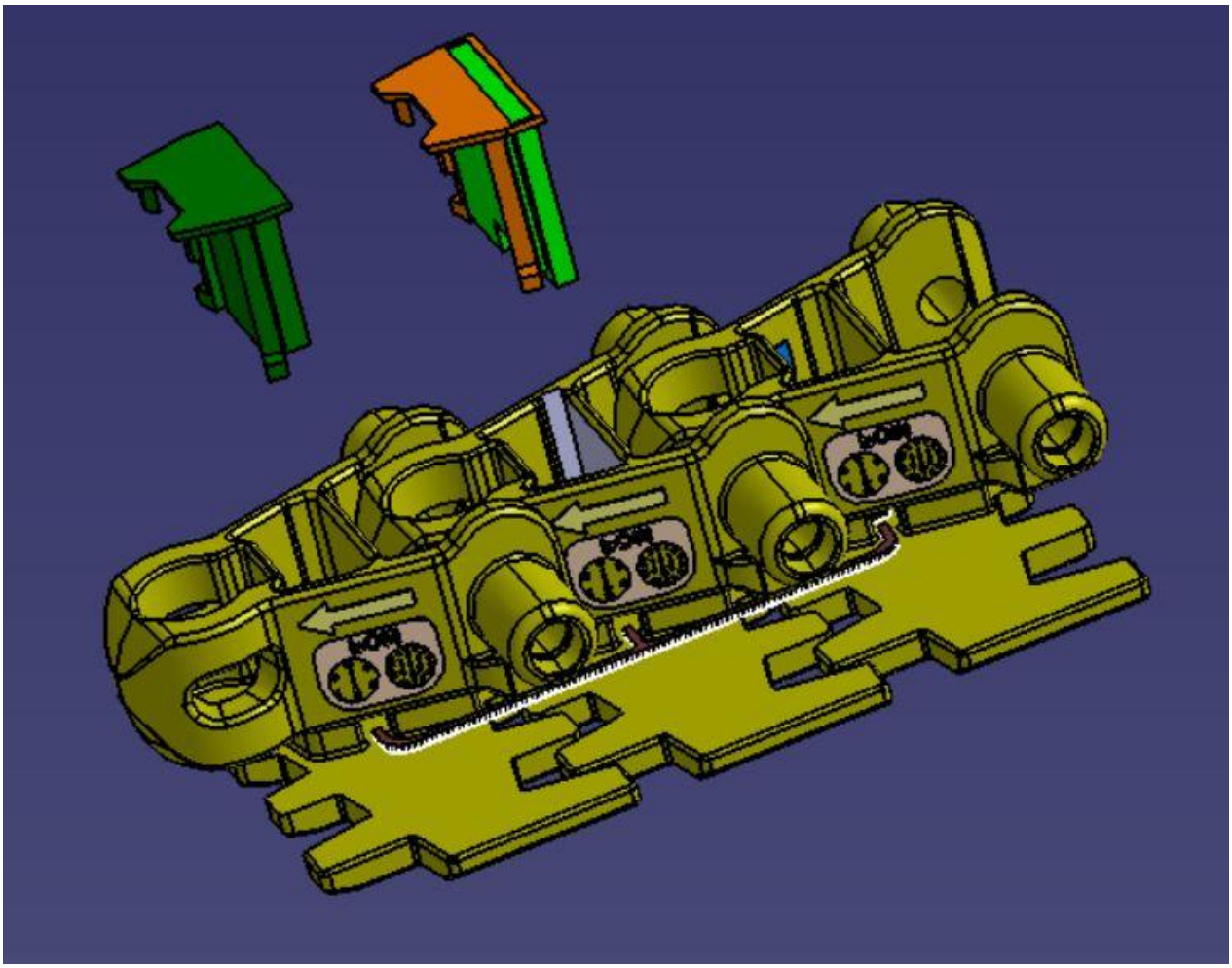
CHALMERS



Smart Link

Trådlös plattform för datainsamling på conveyor band

Examensarbete inom högskoleingenjörsprogrammet Mekatronik

EMIL SOLBERG
KARL WIKHOLM

Institutionen för Elektroteknik
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2018

Förord

De som utfört examensarbetet är Karl Wikholm och Emil Solberg som läser Chalmers tekniska högskolas högskoleingenjörsprogram med inriktning mot mekatronik. Examensarbetet är utfört på företaget Flexlink i Göteborg som i huvudsak tillverkar och utvecklar transportband till industri. Syftet med examensarbetet är att undersöka om det finns möjlighet att trådlöst samla in mätdata från ett band i rörelse och via de mätdata som ges skapa en ökad förståelse för bandets dynamik samt att kunna identifiera underhåll.

Vi vill tacka Jonas Rahm för möjligheten och förtroendet han gett oss att genomföra projektet för Flexlink. Ett tack till Jerry Dahlgren som delgett oss viktig information och hjälp vid tester. Ett tack till Fred Watrous för hjälp med mjukvaruutveckling.

Vi vill även tacka Bertil Thomas för hjälpen vid skrivandet av rapporten.

Göteborg, juni 2018, Karl Wikholm och Emil Solberg.

Sammanfattning

Företaget Flexlink med säte i Göteborg vill ha en trådlös givare som kan mäta rörelser på ett transportband. I dagens läge använder sig företaget av stationära vibrationsgivare som kopplas samman med en PC via USB-kabel. Denna metod ger möjligheten att mäta vibrationer i stativen men inga möjligheter att göra mätningar på bandet. Frågeställningen blir då om man kan utveckla en plattform som kan fästas på bandet för att samla in vibrationsdata samt andra typer av data.

De elektriska komponenter som används är i huvudsak ett utvecklingskort från Adafruit med processorn ESP32 från Espressif och en MEMS-sensor från Bosch sensortec. Dessa två komponenter kopplas samman via en I2C-databuss för att skapa en tillförlitlig kommunikation. För att fästa komponenterna på bandet, till att möjliggöra tester, så designas och tillverkas några olika fästanordningar i plast. Sammansättningen av dessa komponenter och fästanordningar utgör en testenhet som kan användas i flexlinks labbmiljö.

Mätdata skickas mellan givare och server via wifi och TCP/IP-kommunikation. Som server finns möjlighet att använda en PC eller företagets databastjänst. För att visualisera och filtrera mätdata används två olika program, Kst då data läses in via PC och Grafana vid användning av databastjänst.

Valet av plattform samt utformning av program är gjorda för att möjliggöra vidareutveckling. Tex att ansluta fler typer av sensorer eller givare.

Med testenheten som grund har ett förslag på en färdig produkt tagits fram i virtuell miljö. En kretskortdesign har skapats där både sensorn och mikrokontrollern sitter på samma kretskort och som har en storlek anpassad till att passa i en av företagets länkar.

Abstract

The company Flexlink, based in Gothenburg, whants to explore wireless sensors that can measure movements and vibrations of the chain on a conveyor belt.

Today the company uses stationary accelerometer sensors that are connected to a PC via USB cable. This method provides the ability to measure vibrations on the conveyor beams but not directly on the chain, due to insufficient space and that it must be wireless. Is there a way to develop a platform that can be attached to the chain to collect vibration data and other types of data without interfering with the process?

The main electrical components used in this project are a development board from Adafruit with ESP32 processor from Espressif and a MEMS sensor from Bosch sensortec. These two components are connected via an I2C data bus to create a reliable communication. To attach the components to the conveyor chain to enable tests, a variety of plastic attachments are designed and manufactured. The build of these components and attachments is tested and used in a Flexlink lab environment.

Sensor data is sent from the unit to a server through Wi-Fi and TCP / IP communication. To act as server and receiver, you can use a PC or Flexlinks database services. To visualize and filter the sensor data, two different programs are used. Kst 2.0.x when sending data to a PC and Grafana when using database services.

The choice of components and program design are made to enable further development, such as connecting more sensors.
Based on the build and tests, a proposal for a finished product has been developed in a virtual environment. A circuit board design has been created where both the sensor and the microcontroller are on the same circuit board and with a size adapted to fit within a chain link.

Innehållsförteckning

Terminologi 1

1. Inledning 2

1.1 Bakgrund 2

1.2 Syfte 2

1.3 Avgränsningar 2

1.4 Precisering av frågeställningen 2

2. Teori/Teknisk bakgrund 3

2.1 Transportbandets uppbyggnad 3

2.2 Hårdvara 3

2.3 Mjukvara 5

3. Metod 6

3.1 Undersökningar 6

3.2 Konstruktion 6

3.3 Tester 6

3.4 Design av prototyp 7

3.5 Analys av resultat 7

4. Undersökningar 8

4.1 Inledande undersökning med VibSensor 8

4.2 Val av hårdvara 12

5. Konstruktion 14

5.1 Krets 14

5.2 Programmering 15

5.2.1 Esp-IDF 16

5.2.2 I2C 16

5.2.3 Wifi 16

5.2.4 TCP/IP 17

5.2.5 Kommandotolk 19

5.2.6 Main 20

5.3 Testprototyp 20

6 Tester 24

6.1 Datainsamling 24

6.2 Tracking 26

7 Produktprototyp 28

8 Resultat 31

9 Slutsats/Diskussion 32

9.1 Vidareutveckling 32
Referenser 33
Bilaga A: Flödeschema för program ..... A
Bilaga B: Kretskort ..... D

Terminologi

UART = Protokoll för seriell dataöverföring.
I2C = Protokoll för dataöverföring.
SPI = Protokoll för seriell dataöverföring.
SDA = Minnesadressering för I2C operation.
SCL = Klockan för I2C operation.
BNO055 = Sensor bestående av accelerometer, gyro och magnetometer.
BLE = Bluetooth Low Energy, även kallat bluetooth 4.0.
TCP = Transmission Control Protocol (Dataöverföringsprotokoll)
UDP = User Datagram Protocol (Dataöverföringsprotokoll)
AHRS = Augmented Heading and Reference System (Tracking algorithm)
IMU = Inertial Measurement Unit
API = Application Programming Interface
NVS = Non-Volatile Storage
RTOS = Real Time Operating System
ACC = Accelerometer
Gyro = Gyroskop
ESP32 = Microcontroller
AD = Omvandling från analog till digital
DA = Omvandling från digital till analog
PCB = Printed Circuit Board
GND = Ground (Jord)
MEMS = MicroElectroMechanical Systems
Bash = En typ av kommandotolk

1. Inledning

1.1 Bakgrund

Flexlink är ett företag med bas i Göteborg som tillverkar och utvecklar transportband bestående av kedjor av hårdplast. På företaget finns en testavdelning som gör olika typer av utredningar och tester kring bandets funktionalitet och dynamik. I dagens läge har de endast tillgång till att samla in accelerometerdata från stationära punkter. Företaget har uttryckt en önskan att kunna samla in, i huvudsak accelerometerdata, relaterat till kedjans dynamik vid drift. Behovet finns även att kontinuerligt samla in och övervaka olika typer av data från ett band i produktion.

1.2 Syfte

Syftet med projektet är att undersöka om man på ett bra sätt kan samla in mätdata i ett trådlöst och mobilt gränssnitt anpassat efter företagets produkt. Projektet syftar även till att mäta vibrationer och rörelser trådlöst samt att lagra datapunkter i en databas. Där en prototyp ska utvecklas och visualiseras, med hjälp av en 3D-modell, för att visa hur en färdig produkt skulle kunna se ut.

1.3 Avgränsningar

Arbetet syftar inte till att leverera en färdig produkt utan snarare en prototyp och en ide kring hur den slutgiltiga produkten kan se ut. Det syftar heller inte till att bygga en databas då företaget har en redan tillgänglig databasstruktur.

1.4 Precisering av frågeställningen

Grundförutsättningarna är att överföra data via ett trådlöst protokoll såsom Wifi eller Bluetooth. Mätdata ska samlas in och skickas med en uppdateringsfrekvens anpassad till det frekvensspektra som företaget är intresserade av.

2. Teori/Teknisk bakgrund

2.1 Transportbandets uppbyggnad

Bandets mekaniska konstruktion består av två huvuddelar. Den ena delen är ramverket som fungerar som fäste för drivanordningen. Ramverket är oftast en sammansättning av olika aluminiumprofiler, men finns även i rostfritt utförande. Sammansättningen görs med hjälp av olika typer av skruvförband. Motorn som driver transportbandet är en asynkronmotor som oftast sitter monterad längst ut på anordningen. Drivaxeln till motorn är monterad på ett kugghjul i plast som har samma delning på kuggarna som avståndet mellan två länkar i bandet.

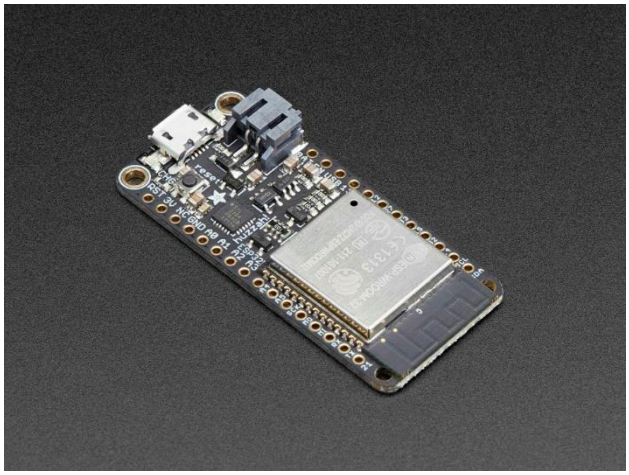
Figur 1: Rak conveyor från Flexlink.

Transportbandet är uppbyggt av ett godtyckligt antal länkar. Länkarna finns i flera olika storlekar och tillverkas oftast i ett slitstarkt plastmaterial, men det finns även hybrider med vissa delar i metall. Eftersom att bandet är uppbyggt av länkar får det samma karaktäristik som en drivkedja, bandet kallas även ofta för plastkedja.

2.2 Hårdvara

Utvecklingskort:
Huvudenheten som används i projektet är ett utvecklingskort från Adafruit med beteckningen "ESP32 Feather Board". Kortet består av en processorenhet (ESP32) från Espressif som har egenskaper som passar projektets kravspecifikation. ESP32 har en inbyggd Wifi-modul och bluetooth-modul (BLE) för trådlös kommunikation. Seriell dataöverföring är möjlig via I2C, SPI eller UART. Det finns en dubbelkärnig processor som har en maximal klockhastighet på 240 MHz. Enheten tillhandahåller också 12st I/O-portar med AD/DA-omvandling som kan användas för ingående och utgående signaler.

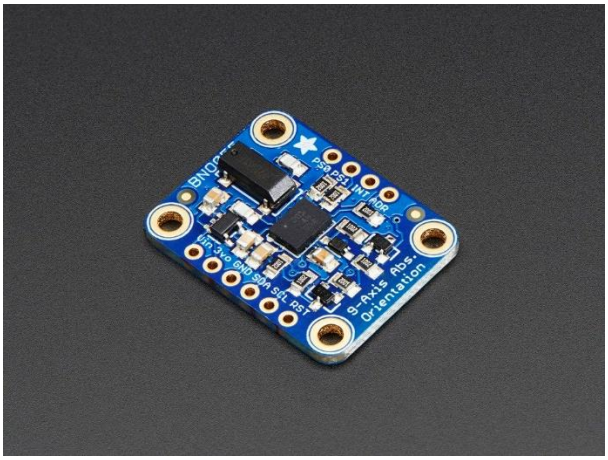
Kortet har möjlighet att ansluta ett batteri för strömförsörjning via en standardkontakt med spännning (3.3V) och GND. Enheten kan även drivas via micro-usb (5V) som också används för att kommunicera med enheten via USB till UART protokoll. [1]



Figur 2: Utvecklingskort från Adafruit. [2]

Sensor:

Den givare som används för att samla in mätdata är en BNO055 från Bosch Sensortec som är en MEMS-sensor med inbyggd accelerometer, gyrometer och magnetometer. Denna enhet kommer också med en integrerad mikroprocessor som kan med hjälp av datapunkterna kan beräkna IMU-mätpunkter och eulerangles. Eftersom att enheten är väldigt liten så har en breakout-board använts för att göra det möjligt att konstruera en krets för hand. Den variant som använts är adafruits breakout board som är utrustad med pull-up motstånd för att säkerställa en säker anslutning. Det kommunikationsprotokoll som erhålls är I2C-protokollet. Enheten drivs av en spänning på 3,3V. [3]



Figur 3: Breakout board från adafruit med sensorn BNO055 monterad i mitten. [4]

2.3 Mjukvara

Eclipse:
Ett open-source program [5] som är en programmeringsmiljö för C code, som underlättar utformning och felsökning av program. Med implementering ut av programbiblioteken: ESP32 - IDF v3.0 [6] en SDK för ESP32 i C code samt freeRTOS [7].

Kst 2.0.x Rev. 0c4519e:
Kst plot är ett open source program som möjliggör plottning av data, samt filtrering.
Kst kan läsa och plotta data live, det vill säga läsa in nya datapunkter med ett tidsintervall på 10ms. [8]

Matlab R2017b:
Har använts till att visualisera insamlade datapunkter samt till att filtrera och försöka realisera objektets bana i ett 3D-rum.

Wireshark 2.4.5:
Programmet Wire shark används till att analysera nätverksprotokoll och låter dig se vad som skickas på ditt nätverk. Programmet har använts till att felsöka TCP/IP kommunikation mellan server och client. Programmet är av typen open source. [9].

Terminal macOS:
Med Terminalen kan vi kommunicera med ESP32 via seriell monitor (USB) samt skapa en server som lyssnar efter klienten. Servern kan ta emot data och med hjälp av terminalen kan vi lagra datapunkterna till en fil. Terminalen i OS X är Linux baserad och Bash-kommandon används.

MINGW32:
En Linux baserat bash-kommandotolk för Windows, fungerar på samma sätt som terminalen för macOS. [10]

Grafana:
Ett molnbaserat analysverktyg för visualisering av data, genom plottning, samt visa på trender och utmärkande data. [11]

Rabbit MQ:
Message broker som används av företaget. [12]

Vibsensor:
Vibsensor är en iOS applikation som samlar in data från den inbyggda accelerometern i telefonen och plottar den på telefonen. Möjligheten finns även att dela datapunkterna till matlab. Max uppdateringsfrekvens är 100Hz. [13]

Catia V5:
Catia är en CAD programvara som används för att designa produkter i en 3D-miljö. Programmet kan även användas för att virtuellt montera färdiga detaljer till en produkt.

3. Metod

3.1 Undersökningar

- **Systemets uppbyggnad och funktionalitet**
  Ta reda på hur transportbandet fungerar samt att ta reda på hur det används.

- **Tidigare undersökningar inom området**
  Finns det tidigare gjorda tester och forskning inom området som gjorts av företaget, och finns det andra aktorer som behandlat liknande studier.

- **Intervjuer av medarbetare**
  Ta reda på vilken vetskap som finns inom företaget och om det finns ej provade hypoteser.

- **Hitta relevanta mätmetoder för analys av bandets dynamik**
  Utifrån den information som införskaffats ska några mätmetoder utvecklas så att bandets dynamik kan avläsas.

- **Val av mätutrustning**
  Vilken mätutrustning behövs för att kunna testa de mätmetoder som utvecklats.

3.2 Konstruktion

- **Kommunikation mellan huvudenhet och sensor**
  Upprätta ett kommunikationsprotokoll för överföring av mätdata, samt skapa drivrutiner för sensorn.

- **Trådlös kommunikation mellan huvudenhet och server**
  Säkerställa en stabil uppkoppling och överföring utav mätdata.

- **Design av testprototyp**
  Konstruktion av behållare för huvudenhet och sensor. Så att dessa kan fästas på kedjan utan att störa driften. Samt integrering av indikatorlampor och konfigurationsknapp.

3.3 Tester

- **Test av nätverksöverföringens prestanda**
  Tester för att säkerställa att överföringshastigheten är tillräckligt snabb.

- **Insamling av accelerometerdata**
  Säkerställa att accelerationsdata samlas in i den hastighet som angetts.

- **Test av AHRS algoritm**
  Testa möjligheten att implementera en AHRS-algoritm för att spåra givarens position och för att försöka skapa en virtuell 3D-modell av bandet.

3.4 Design av prototyp

- **Bestämma storleken på kretskortet**
  Ta reda på hur stort kretskortet med alla komponenter kan vara för att få plats i en länk. Samt identifiera vilka komponenter som krävs.

- **Konstruera ett kretskort**
  Konstruera ett kretskort som får plats i länken.

- **Skapa en ett fäste för kretskortet**
  Skapa en fästanordning för kretskortet som håller kortet på plats i länken.

3.5 Analys av resultat

- **Validering av resultat**
  Stämmer resultatet överens med kravspecifikationen?

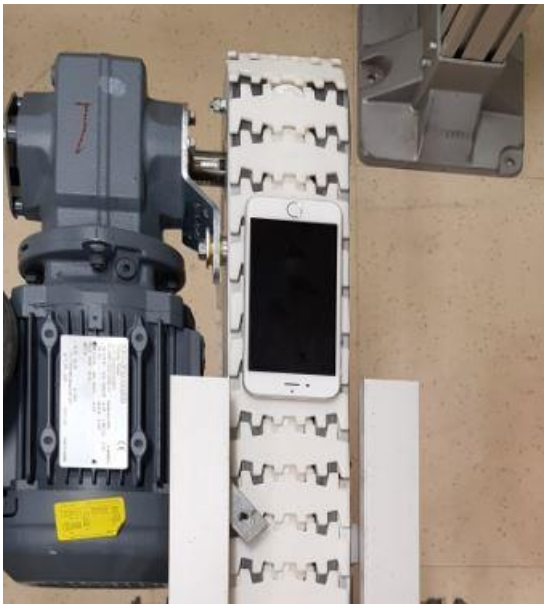
- **Diskussion kring vidareutveckling**
  Resonera kring hur man kan gå vidare med produktprototypen för att skapa en färdig produkt.

4. Undersökningar

I det här kapitlet avhandlas det förberedande arbetet som leder fram till att en prototyp konstrueras och realiseras.

4.1 Inledande undersökning med VibSensor

Som ett initialt test av frågeställningen användes applikationen VibSensor. En mobiltelefon av modellen iPhone 6s, där applikationen är installerad, åker med runt på ett transportband i flexlinks testlabb. Den sensor som är integrerad i mobiltelefonen har en uppdateringsfrekvens på 100Hz. Applikationen är programmerad så att den ska samla in accelerometer och gyro data från mobiltelefonens interna accelerometer. Data sparas ner på telefonens interna minne så att man efter testet kan överföra mätpunkter till en dator. Vidare kunde mätpunkterna överföras till Matlab där vidare analys gjordes.



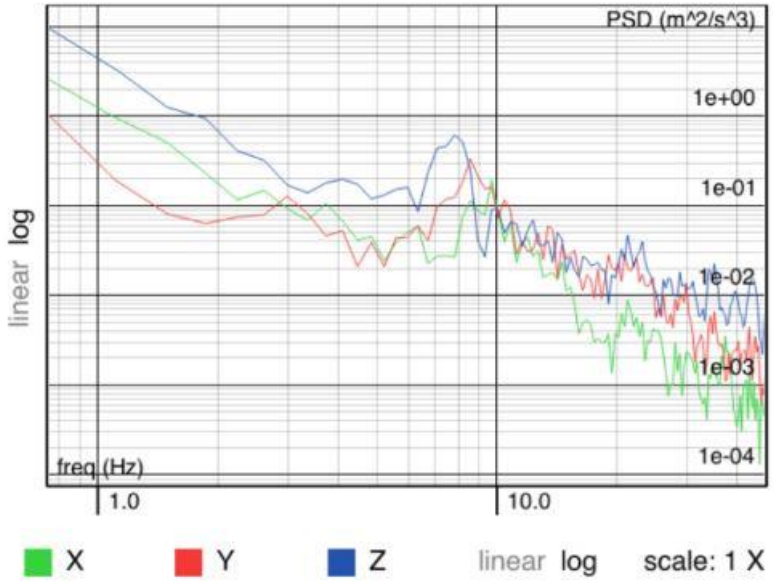
Figur 4: Bilden beskriver hur telefonen placerades på bandet.

Målet med att använda VibSensor är att initialt få en bild kring hur mätdata ser ut när man låter en accelerometer åka med bandet. Intresset ligger mycket kring vilket frekvensspann som krävs för att mäta de rörelser och vibrationer som uppstår i bandet under drift. Dessa initiala mätningar kan då användas som underlag för val av hårdvara. Även att undersöka om det redan i ett initialt skede går att urskilja någon effekt som är av intresse.

Det första testet gjordes på ett stabilt transportband med enkel utformning. Bandet bestod endast av en raksträcka som var en meter lång. Länkarna som användes till detta band var av typen X85 som är en av de större länkarna i Flexlinks sortiment. Detta betyder att bandets bredd är 85mm. Motorn som driver bandet är en asynkronmotor där hastigheten på bandet ställs in med intervallet 0–100% vilket motsvarar andel av full effekt till motorn.

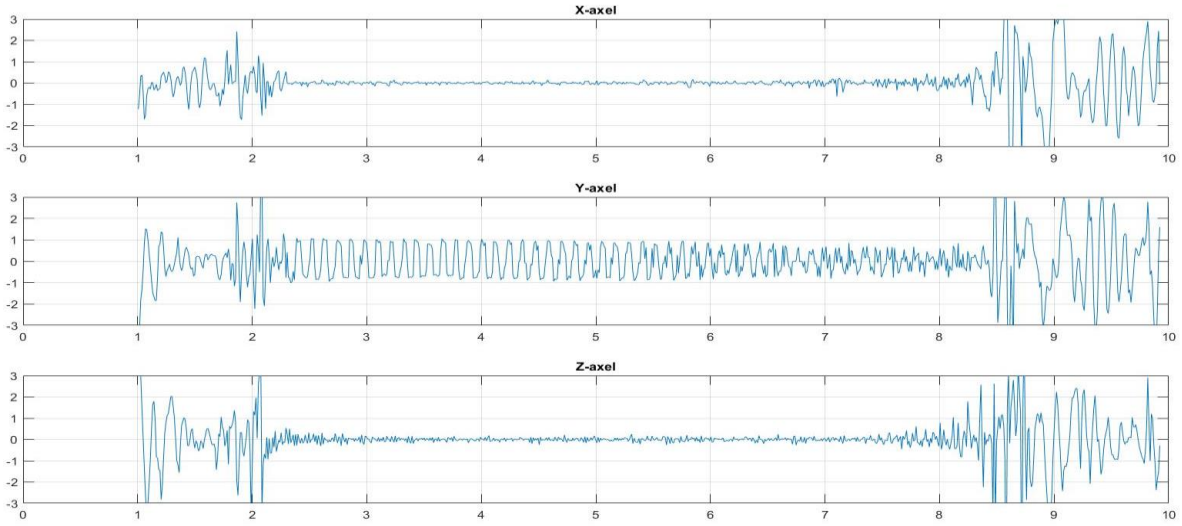
Figur 5: Bandet som första testet utfördes på.

När testet är slutfört används den inbyggda funktionen för FFT-analys i applikationen för att försöka urskilja vilka frekvenser som är intressanta. Som man kan se i bilden nedan är det svårt att urskilja om det är någon frekvens som sticker ut och är mer relevant än de andra. Ett test med en givare som registrerar data med bredare frekvensspann måste utföras för att få mer fullständig information.



Figur 6: FFT-analys för rakt transportband.

Sedan skickas de mätvärden som samlats in till en dator för vidare analys i matlab. Punkterna för vardera axeln plottas i varsin ruta för att försöka urskilja några mönster. I x-axeln och i z-axeln kan man inte urskilja något annat än brus. Men i y-axeln kan man se ett mönster som har en återkommande periodtid. Detta fenomen skulle kunna vara polygoneffekten som ofta uppstår i system som är drivna av ett kugghjul. Ett kugghjul med tänder får ett rörelsemönster som liknar en polygon vilket gör att kraftöverföringen mellan kedjan och kugghjulet varierar med en period som korresponderar med delningen av kuggar på kugghjulet. De höga toppar som finns i början och i slutet för varje axel tillkommer då mobiltelefonen läggs på bandet.

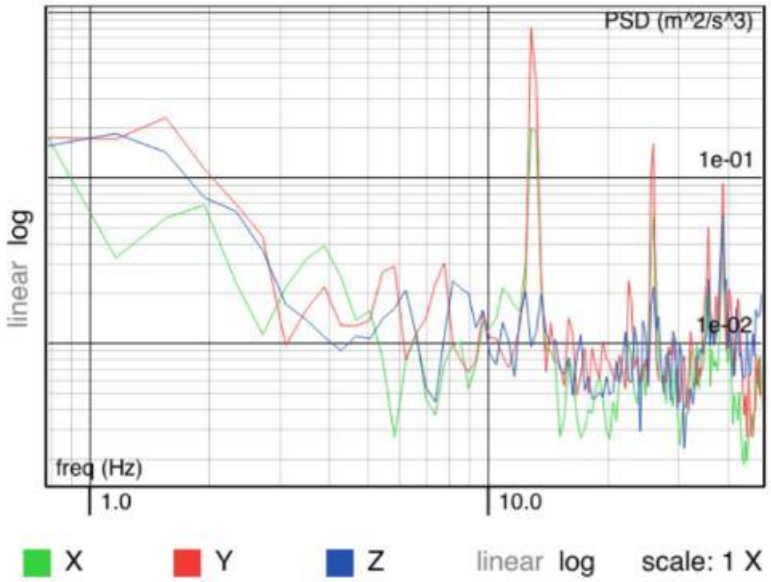


Figur 7: Initialt test på rakt transportband.

Det andra testet som utfördes var på ett annat transportband som var installerat för att vara instabilt och som även har en annan bana. Detta transportband har en bana som består av två raksträckor och en 90 graders kurva. Bandet är även belastat med maximal last för detta band, även det bidrar till en instabil drift. Det drivs på samma sätt som bandet i test ett.

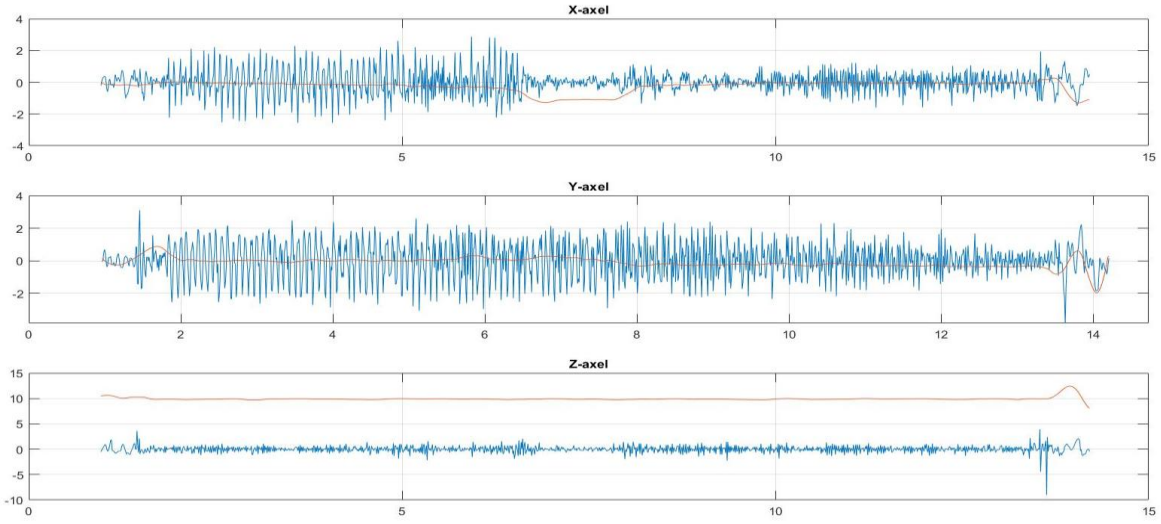
Figur 8: Instabil, högt belastad conveyor med 90 graders kurva.

En FFT-analys gjordes även på detta transportband. Här kan man urskilja några frekvenser som är intressanta. Man kan se att det finns signifikanta toppar på frekvenserna 14 Hz och 26 Hz. Dessa toppar beror förmodligen på att transportbandet står instabilt och gungar med en viss frekvens. Den informationen tas vidare till konstruktionsfasen.



Figur 9: FFT-analys för bana två.

I det andra testet är det tydligt att transportbandet är instabilt. Amplituden på signalen är högre och rörelserna är mer högfrekventa. I detta test visualiseras både accelerometer och gyrodata för att kunna påvisa relationer mellan acceleration och vinkelförändring. De första sju sekunderna visar när telefonen färdas på raksträcka ett. I tidsintervallet 7–9 sekunder så färdas den i kurvan och resterande tid är när den färdas på raksträcka 2 (se figur 8). När man kollar på tidsintervallet 7–9 sekunder kan man se att man i x-axeln får en vinkelförändring från gyrodata och man kan även se att vibrationerna minskar när den åker genom kurvan. Detta är något som man skulle kunna använda för att ta reda på givarens nuvarande position på bandet.



Figur 10: Test på ett instabilt transportband med 90 graders kurva. Blå kurva representerar accelerationsdata och orange kurva representerar datapunkter från gyroskop.

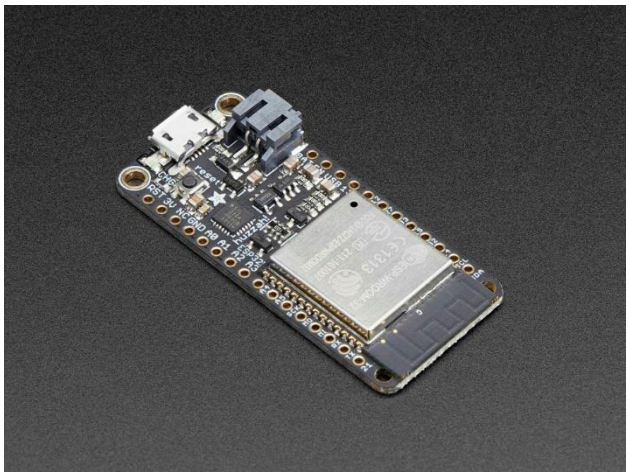
Sammanfattningsvis, det finns några saker att ta med sig till val av hårdvara. För att vara säker på att uppdateringsfrekvensen på 100Hz är tillräckligt hög behövs fler tester med en enhet som har högre uppdateringsfrekvens. Det kan även vara intressant att veta vilken position som givaren har på bandet för att lättare kunna analysera datapunkterna.

4.2 Val av hårdvara

Med tidigare undersökningar som grund väljs hårdvara som uppfyller kraven. Kraven är att uppdateringsfrekvensen ska vara högre än 100Hz och att det ska vara möjligt att mäta både vinkelacceleration och acceleration. Man ska även kunna skicka de insamlade datapunkterna i realtid till någon typ av databas. För att göra det krävs en trådlös överföring i form av bluetooth, wifi eller annat alternativ. En annan viktig aspekt är att enheten ska vara energisnål eftersom att den kommer att vara mobil och kan därför inte kan vara fast ansluten till en strömkälla. Den ska kunna hantera seriell kommunikation och den ska även kunna hantera AD/DA omvandling. En annan viktig sak är storleken på komponenterna då länkarna i transportbandet varierar i storlek och vissa kan vara väldigt små så vill man gärna att enheten ska passa alla länkar.

Valet av huvudenhet faller på en enhet från Esspressif systems. Enheten heter Esp32 och är en ny utvecklingsenhet med bra egenskaper. Den har WiFi och BLE för trådlös kommunikation och den hanterar SPI, I2C, I2S och UART för seriell kommunikation. Den har flertalet in/utgångar som kan hantera både AD och DA omvandling vilket gör den till en bra plattform för ändamålet men även för vidareutveckling.

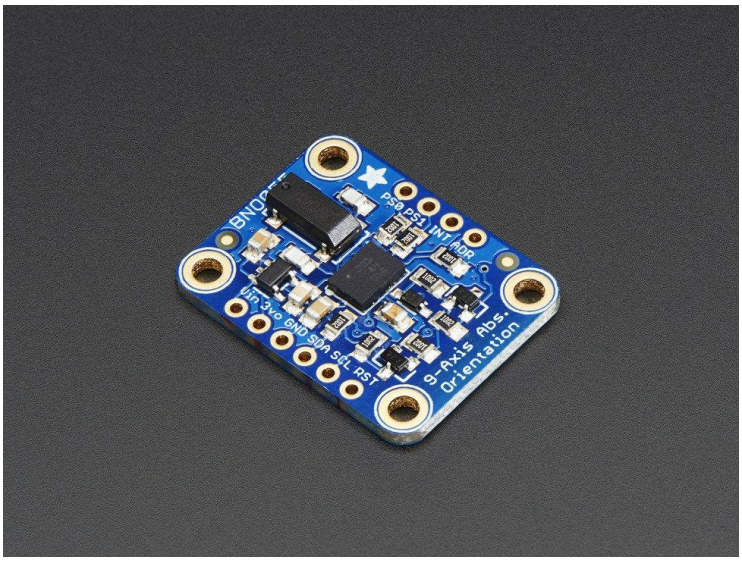
En stor fördel med att använda enheten esp32 är att den finns i flera olika utföranden vilket är väldigt förmånligt då planen är att integrera enheten i en länk där det råder stor brist på utrymme. Det utförande som används i utvecklingsarbetet är ett utvecklingskort från Adafruit Industries som kallas för Adafruit HUZZAH32. Strömförsörjningen kommer initialt ske via ett litiumbatteri med arbetsspänning på 3,3V och en kapacitet på 250mAh.



Figur 11: Utvecklingskort från Adafruit. [2]

Vid val av sensor togs beslutet att köpa en BNO055 från Bosch sensortec. Den köptes in monterad på ett utvecklingskort, även kallad breakout-board, designat av Adafruit Industries.

Fördelen med denna sensor är att den har möjlighet att kommunicera över I2C-protokollet vilket är kompatibelt med huvudenheten. Sensorn har flertalet intressanta egenskaper. Den har inbyggd accelerometer som kan samla in mätpunkter med en maximal uppdateringsfrekvens på 1000Hz, ett gyroskop med en maximal uppdateringsfrekvens på 500Hz och en magnetometer med en uppdateringsfrekvens på 20Hz. Den har även en inbyggd mikroprocessor som ger möjligheten att läsa ut datapunkter av typen IMU, eulerangels eller NDOF. Dessa typer av mätpunkter kan användas för att beräkna positionen för sensorn vilket är ett önskemål för att lättare kunna analyserna mätpunkterna.



Figur 12: Breakout board från adafruit med sensorn BNO055 monterad. [4]

5. Konstruktion

I det här kapitlet avhandlas hur arbetet från en första krets till en färdig testprototyp har gått till. Det hanterar också hur enheten är programmerad och uppkopplad till ett nätverk.

5.1 Krets

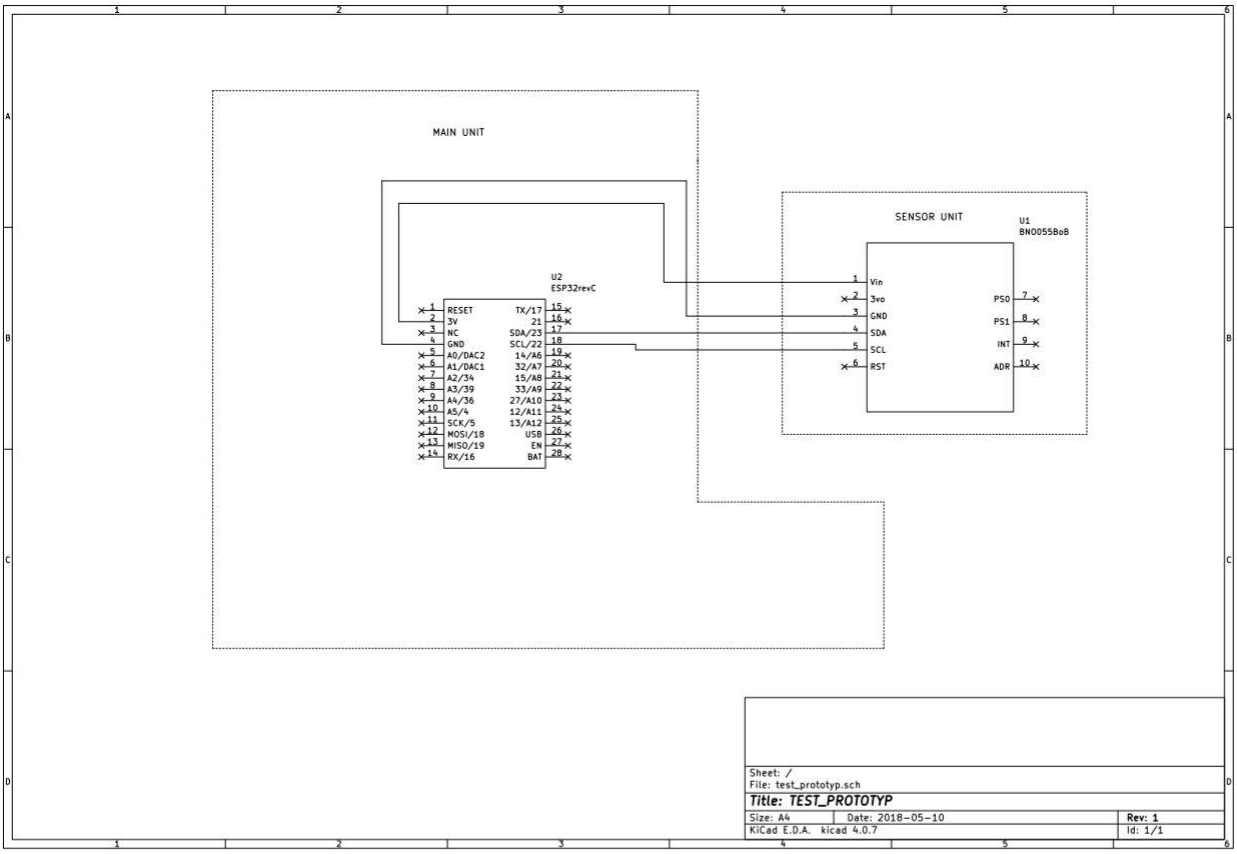
Det första kretsbygget är utformat utifrån att kommunikationsprotokollet I2C används som kommunikationsbuss mellan huvudenheten och sensorn. De båda komponenterna stödjer protokollet och en busshastigheten på 400kHz är tillräcklig för önskat informationsflöde. Huvudenheten agerar “master” och sensorn “slave” där mastern styr kommunikationsflödet och läser i från slavarna. Med I2C-protokollet kan flera slavar kopplas samman i serie och kommunicera på samma ledare.

Både sensorn BNO055 och mikrokontrollern ESP32 är monterade på varsitt utvecklingskort, vilket medför att de är väl förberedda för att kopplas samman, följande kopplingar behövs för att kommunikation dem emellan ska vara möjlig.

Sensorn matas med spänning från huvudenheten, två kopplingar: 3,3V och GND. För I2C kommunikation krävs två kopplingar: SDA (dataöverföring) och SCL (klockfrekvens). Det betyder fyra ledare mellan huvudenheten och sensorn.

SDA och SCL har varsitt pull-up motstånd monterat på sensorns utvecklingskort, på vardera 10kHz, detta för att förhindra kommunikationsavbrott.

Huvudenheten har en USB-port med UART-kommunikation, vilket möjliggör att skriva program till minnet samt att kommunicera via en seriell monitor från en PC.



Figur 13: Kretsschema för test av i2C kommunikation.

Kretsen är kopplad på ett experimentkort med banor. Kortet utformades inte för att fästas på en flexlink-bana utan endast till att möjliggöra tester av I2C-kommunikation samt utveckling av drivrutiner för sensorn.

5.2 Programmering

Kontrollenheten ESP32 kan använda sig av sin ursprungliga programmeringsplattform, eller med en Arduinoplattform. Valet för projektet föll på att behålla den ursprungliga plattformen och att programmera i C. Anledningen är att, även om arduino är en enklare och i viss mån en snabbare utvecklingsmiljö, så har den också begränsningar. De programbibliotek, API:er, som finns i Arduino-miljö är inte alls lika många och det finns inte samma enkelhet att specialanpassa programmet för ändamålet.

Ytterligare anledningar är att vid vidareutveckling så är en C miljö enklare att standardisera och möjlighet finns att använda de senaste API:erna ifrån Espressif, då det tar tid innan samma funktioner är kompatibla med arduino. Förutsatt att du ej själv vill skapa den implementeringen till arduino.

Att programmera i C bidrog till en längre utvecklingstid, då mycket arbete gick till att skapa drivrutiner och till att lära sig Esp-idf:funktionerna. Målet var att utveckla en plattform att bygga vidare på och då blev valet ESP-IDF trots vetskapen om en längre utvecklingsperiod.

5.2.1 Esp-IDF

Programmeringsplattformen Eclipse används tillsammans med ESP-IDF.
ESP-IDF är ett utvecklingsverktyg till ESP32-plattformen, som består av API:er och funktioner skrivna i C-kod, framtagna för att förenkla programutveckling och möjliggöra användning av de funktioner som finns hos ESP32. Viktiga API:er för det här projektet är: WIFI, TCP/IP och I2C.

ESP-IDF implementeras i en Linux-miljö där man via en kommandotolk kan kommunicera via USB/UART med så kallade "make commands". Dessa kommandon kan vara *"make monitor"* för att aktivera en seriell monitor, eller *make flash* för att skriva över program till enhetens minne. Monitorn gör det möjligt att övervaka programmet och att kommunicera med enheten.

Kommandot: *make menuconfig* kallar på ett program (meny i kommandotolken) där det går att ställa in konfigurationer för hårdvaran och andra grundläggande inställningar, såsom inställningar för RTOS, UART, med mera. Konfigurationerna sparas till en fil, *sdkconfig,* som används vid kompileringen av programmet. sdkconfig filen går ej att skriva till via Eclipse.

Ett antal färdigskrivna program testades för att säkerställa att hela kedjan för byggandet av program och att kommunicera med enheten fungerade.

5.2.2 I2C

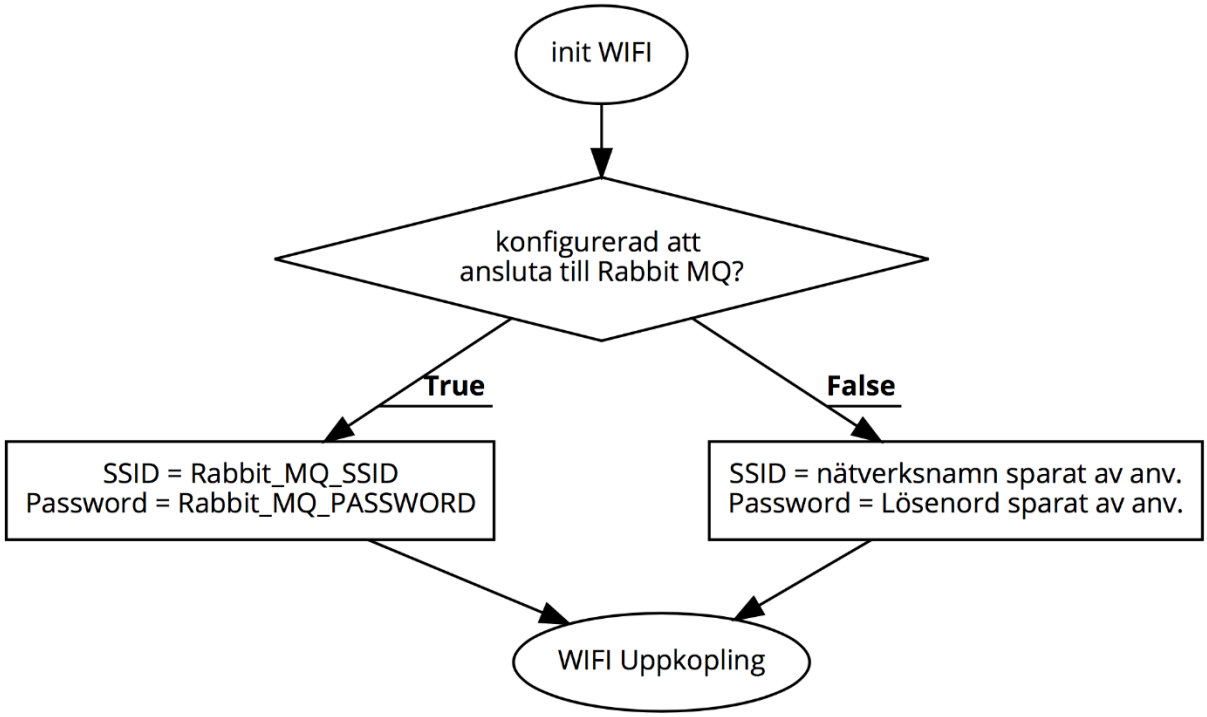
För att kommunicera med accelerometern (BNO055) så används kommunikationsprotokollet I2C. Först krävs en initiering ut av I2C, där man talar om för huvudenheten att I2C ska användas samt hur den ska vara konfigurerad, så som busshastighet och "clock stretching". "Clock-stretching" är en möjlighet att vänta in slaven, istället för att gå vidare vid utebliven respons. Vid de första försöken gick det ej att upprätta någon kommunikation till sensorn, men det visade sig att BNO055 är i behov av en viss "clock-stretching", då den har en egen mikrokontroller som vid tillfällen kräver tid för beräkning.

För att kunna adressera till slaven så behöver mastern ha tillgång till slavens ID-nummer. Slavens ID-nummer finns antingen att hitta i databladet, eller via en sökning på bussen. Efter säkerställd kommunikation, skapades drivrutiner för att kunna konfigurera och läsa data från sensorn.

5.2.3 Wifi

Till att upprätta WIFI-kommunikation används en WIFI-API ifrån ESP-IDF, där in parametrar blir SSID och lösenord. Enheten är konfigurerad till att vara i stationärt tillstånd. WIFI uppkoppling hanterar WPA samt WPA2 kryptering.

Wifi initieras enligt följande:



Figur 14: Flödesschema som visar initiering av WiFi.

5.2.4 TCP/IP

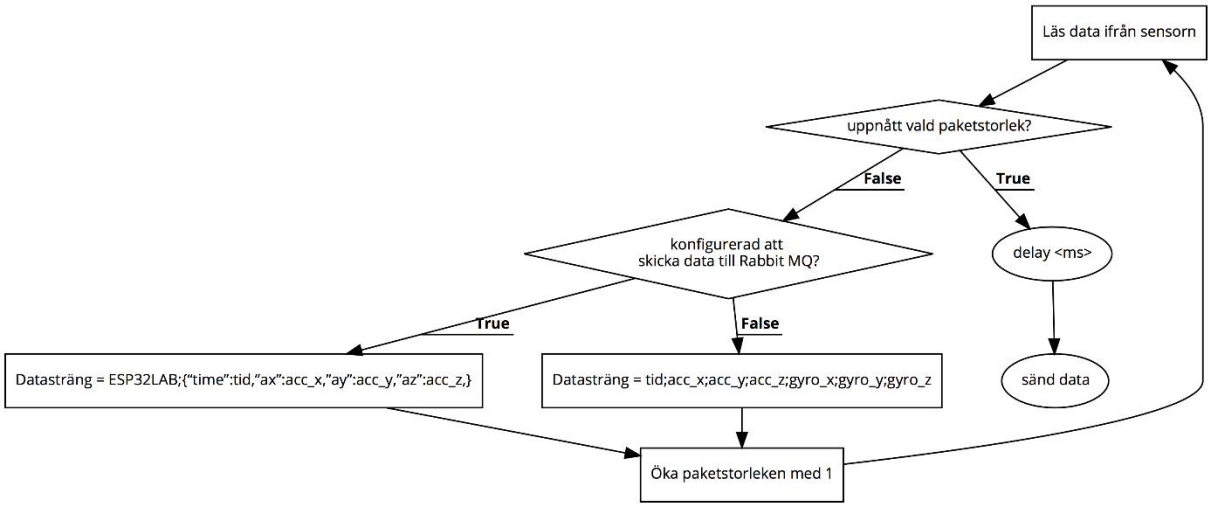
Kommunikationsprotokollet TCP används till att sända data mellan huvudenheten (klient) och en server. Till en början användes en PC för att skapa en server, detta för att säkerställa uppkoppling och överföring av data. Till att utvärdera och analysera kommunikationen används programmet Wireshark, som kan visa mottagna och skickade paket. Wireshark kan identifiera uteblivna paket samt belysa bristerna i kommunikationen.

TCP valdes över UDP då TCP medför att data kommer i rätt ordning och att ej mottagna paket skickas på nytt. Begränsningen blir istället något långsammare en UDP. Vid krav på hög uppdateringsfrekvens i realtid kan kanske UDP passar bättre men på bekostnad att datapunkter kan tappas.

TCP	UDP
Håller koll på borttappade paket, ser till att paket som förloras skickas igen	Håller inte koll på borttappade paket
Ser till att paket kommer fram i rätt ordning och ordnar om paketen i rätt ordning om dom kommer fram i fel ordning.	Bryr sig inte om vilken ordning paket kommer fram i
Långsammare, på grund av alla extra funktioner	Snabbare, eftersom den saknar alla extra funktioner
Exempel på program och tjänster som använder TCP: - HTTP - HTTPS - FTP - Många Datorspel	Exempel på program och tjänster som använder UDP: - DNS - IP Telefoni - DHCP - Många Datorspel

Figur 15: Tabellen förklarar skillnad mellan TCP och UDP. [14]

TCP skickar strängar som tolkas som ASCII kod. Beroende på mottagare så krävs det ett unikt utseende på strängen som skickas. Strängen behöver en "header", ett ID, som visar på vad som skickas och som hjälper servern att hantera informationen och sända den vidare till rätt destination.



Figur 16: Flödesschema för sammanställning av mätdata.

5.2.5 Kommandotolk

För att underlätta användningen av enheten används en kommandotolk via den seriella monitorn. Med kommandotolken är det möjligt att ändra konfigurationer samt få information från enheten.

Följande kommandon finns att tillgå:

Tabell 1: Beskriver kommandon för kommandotolk.

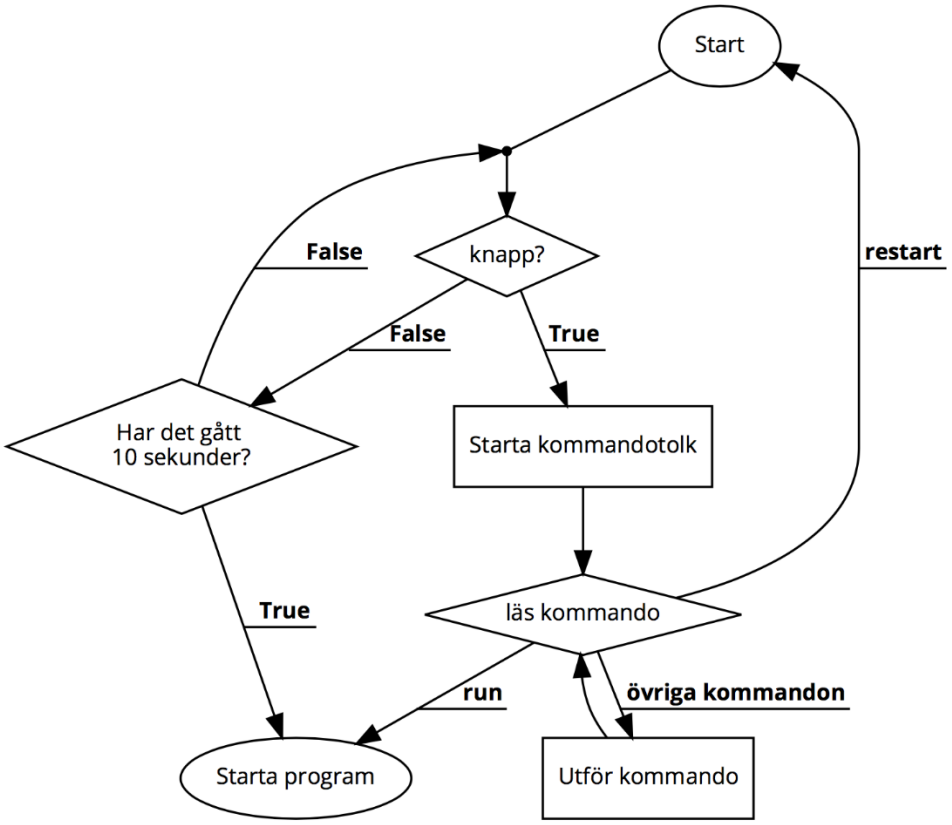
Kommando	Beskrivning	Möjliga in parametrar
help	Listar alla kommandon.	-
run	Starta programmet, (ej omstart)	-
restart	Omstart	-
free	Visa ledigt utrymme på minnet.	-
wifi	Ställ in SSID och lösenord för nätverket du önskar ansluta till.	<ssid> <password>
tcp	Ställ in Serverns IP och port.	<IP> <port>
rabbit	Snabbinställning av wifi och tcp för att skicka data till Rabbit MQ.	<y/n>
opmode	Ställ driftläge för BNO 055.	<1–4> 1 = ACCGYRO, 2 = IMU, 3 = AMG, 4 = NDOF
freq	Ställ uppdateringsfrekvens för accelerometer och gyro. (endast i ACCGYRO och AMG läge, 100Hz fast frekvens vid IMU och NDOF)	<acc/gyro> <Hz> acc: 62,125,250,500,1000 gyro: 12,23,32,47,64,116,230,523 default: acc = 125Hz, gyro = 32dps
range	Ställ omfång för accelerometer och gyro. (endast i ACCGYRO och AMG läge, acc = 2g och gyro = 2000dps är fasta värden vid IMU och NDOF)	<acc/gyro> <g/dps> acc: 2,4,8,16 gyro: 125,250,500,1000,2000 default: acc = 2g, gyro = 2000dps
speed	Ställ in tiden den väntar mellan varje paket skickas. (Hur ofta vill du läsa och skicka data?)	<ms>
p_size	Ställ antal mätpunkter som ska skickas med varje paket.	<1-10> antal mätpunkter.

Inställda värden ska vara sparade i minnet även efter en omstart eller vid strömavbrott. NVS är en API (programfunktion), tillgänglig via ESP-IDF, som hanterar att spara värden och strängar till den del av internminnet som inte nollställs vid omstart.

5.2.6 Main

Vid uppstart av huvudenheten så finns ett fönster på tio sekunder då det går att få tillgång till kommandotolken genom att trycka på knappen. Utan knapptryck startas programmet och enheten börjar med att söka efter en wifi uppkoppling. Då den är uppkopplad mot nätverket tänds den ena lampan och lyser grönt. Nästa steg blir att skapa kommunikation med en server, när kommunikation finns så lyser lampa nummer två också grönt. Skulle uppkopplingarna brytas så lyser lamporna rött istället.

Programmet är uppdelat i två "Tasks" En task hanterar WIFI- och TCP-uppkoppling och en annan som hanterar datainsamling och sändningen ut av datapunkterna.

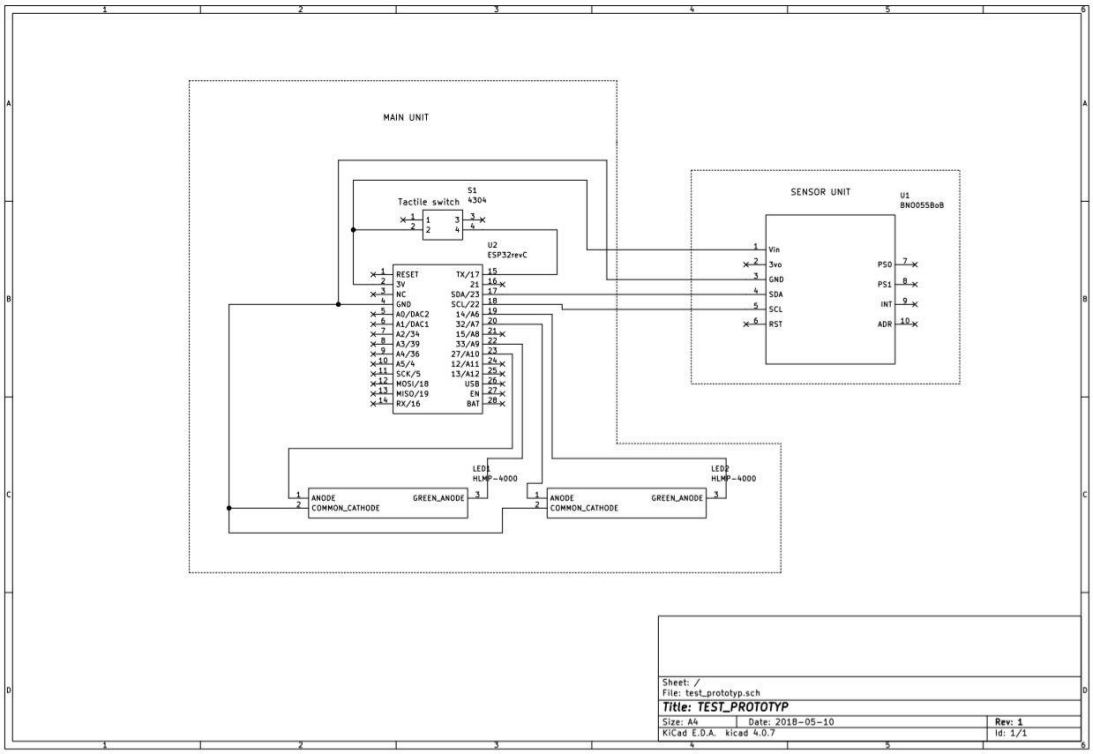


Figur 17: Flödesschema för konfigurering vid uppstart.

5.3 Testprototyp

När första versionen av programmet är utvecklat och testat på kopplingsplattan börjar konstruktionen av en prototyp som kan användas för att göra tester i Flexlinks labbmiljö. Första steget är att göra ett kretsschema för den fullständiga kretsen. Tillägg från det tidigare kretsschemat är att en tryckknapp och två dioder lagts till. Knappen används för att komma in i konfigurationsmenyn för enheten där man kan ställa in nätverksinställningar och välja hastighet för insamling av datapunkter. Det går även att ställa in noggrannheten på de datapunkter som samlas in.

De två dioderna används för att visa status på sensorn och för att se om enheten är uppkopplad mot servern.

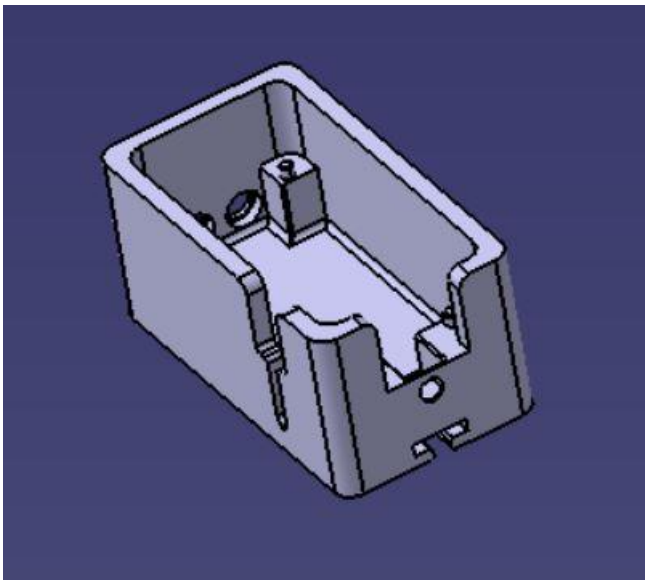


Figur 18: Kretsschema för testprototypen.

För att montera enheten på bandet utvecklas två stycken i olika storlekar i CAD programmet Catia. En skyddande behållare för huvudenheten och en för sensorn. Två olika fästen skapas också för att fungera som ett fäste mellan länken i bandet och den skyddande behållaren.

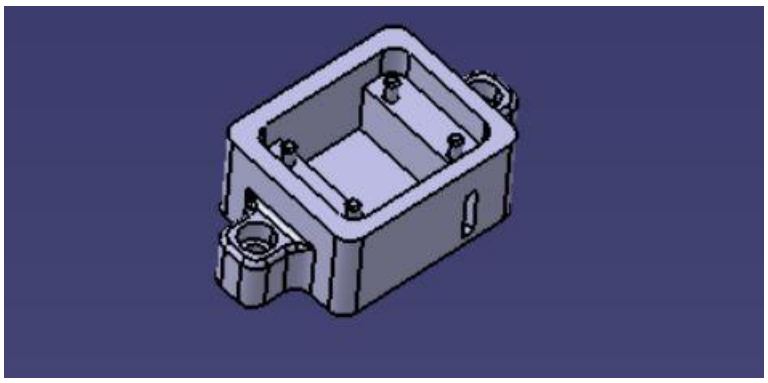
Den stora behållaren designas så att huvudenheten ska kunna fästas på ett bra sätt. Då utvecklingskortet kommer med fyra hål för montering så skapas fyra stycken upphöjda axlar med något mindre diameter än hålen för att skapa en bra passning. Dessa pinnar placeras på 10 mm höga upphöjningar i varje hörn för att skapa utrymme för kablar under kortet. Två hål läggs till på ena kortsidan av behållaren för att kunna fästa de två dioderna. Dessa hål är planförsänkta för att skapa plats för den fläns som finns på dioderna.

På den andra kortsidan skapas ett spår för att få möjlighet att komma åt det USB uttag som finns för att föra över program samt att konfigurera enheten. Men även för att få ett utrymme att montera den knapp som används för att komma in i konfigurations menyn. Ett spår skapas på den ena långsidan för att få möjlighet att koppla in ett batteri som driver enheten. Men även för att göra plats för kablarna som går mellan huvudenheten och sensorenheten som överför data via I2C protokollet. Till sist skapas ett T-spår i botten på behållaren som används för att fästa behållaren i ett fäste.



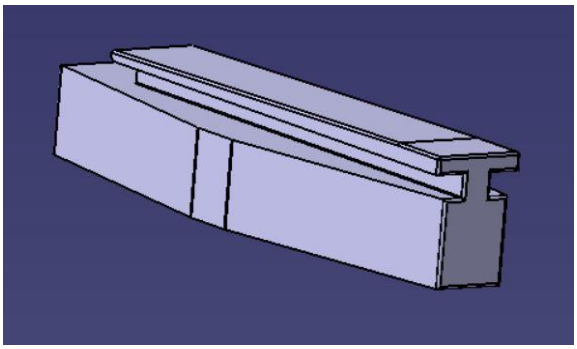
Figur 19: Cad modell av huvudenhetens behållare.

Den lilla behållaren för sensorenheten är designad på liknande vis. Den har invändigt två avsatser som är 10 mm höga och på avsatsen finns fyra stycken axlar som passas i utvecklingskortets fyra hål. Ett avlångt hål läggs till på långsidan för att föra in överföringskablarna från huvudenheten. Två stycken fästöron läggs till med två hål för att göra det möjligt att fästa behållaren i ett fäste. På undersidan finns även två stycken hål som passas ihop med fästet.

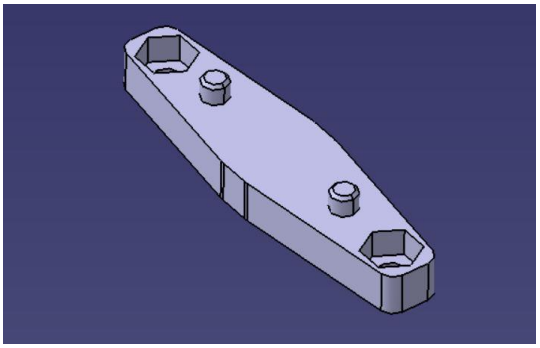


Figur 20: Cad modell av sensorenhetens behållare.

De fästen som används mellan bandet och de olika behållarna har samma basutförande men olika infästning med de olika behållarna. Basutförandet är designat efter länkens utformning. Infästningen som används för att fästa behållaren för huvudenheten är ett T-spår som är kilformat på höjden för att se till så att behållaren kilas fast mellan fästet och botten av behållaren. Fästet utformades på det här sättet för att göra det lätt montera och demontera behållaren. Den behållare som används för sensorenheten har två utskurna polygonala hål där muttrar med M4 standard limmas fast. Sedan använts skruvar med M4 standard som förs genom hålen i öronen på behållaren för att montera behållaren i fästet. Denna Infästningsmetod används för att säkerställa att givaren ligger parallellt med bandet för att inte få några mätfel. Det finns även två stycken fixturpinnar som säkerställer att behållaren monteras vinkelrätt med bandets färdriktning, detta för att inte få några vinkelfel vid montering.



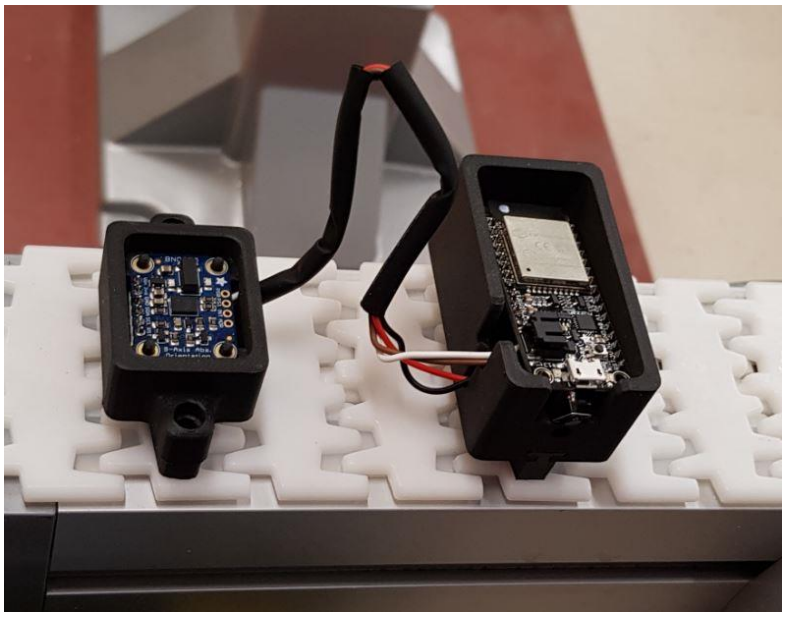
Figur 21: Cad modell av fäste för huvudenhet.



Figur 22: Cad modell av fäste för sensorenhet.

När modellerna är färdiga sparas dom i filformatet STL för att sedan kunna läsas av mjukvaran som används för att generera maskinkod för en 3D-printer. Modellerna printas i Flexlinks 3D-printer som är av modellen Markforged X7 och har en noggrannhet på en hundradels millimeter och printar i lager som är en tiondels millimeter. Maskinen printar med svart plast som är av materialet polyamid som gör att detaljen blir stark och får en snygg finish.

Med alla detaljer realiserade börjar monteringen av de elektriska komponenterna. Kretsen realiseras med hjälp av experimentkort som kapas ut två olika storlekar för att passa de två olika de olika utvecklingskorten. Komponenterna löds fast på de utskurna experimentkorten utefter kretsschemat för testprototypen. Nästa steg blir att löda fast de färdiga experimentkorten på de olika utvecklingskorten. Med hela kretsen färdig monteras kretsen i de två behållarna. Först limmas de två dioderna och konfigurationsknappen i den stora behållaren och till sist limmas och sedan passas utvecklingskorten på de små axlarna som finns i båda behållarna. Med testprototypen färdigmonterad fästs den på bandet och tester av prestanda påbörjas.



Figur 24: Färdig testprototyp monterad på bandet.

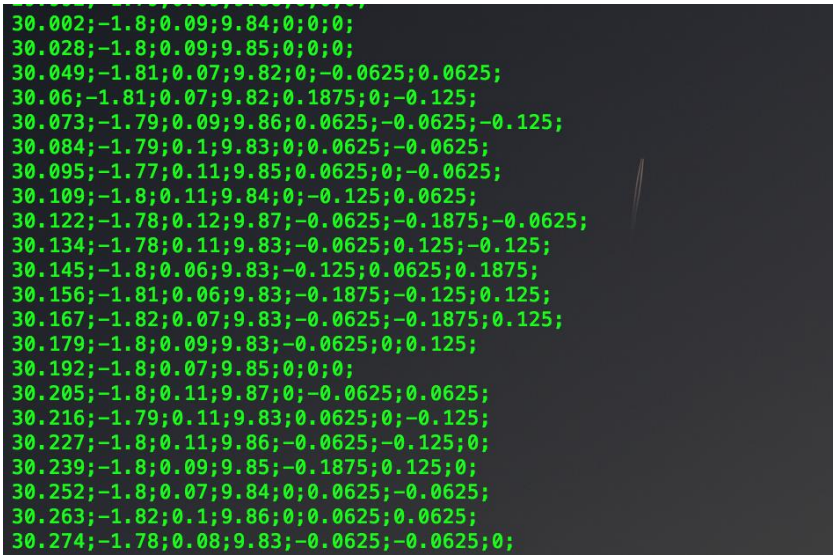
6 Tester

Med testprototypen monterad på bandet kan testerna av prestandan påbörjas. De tester som utförs är test av prestanda för insamling av accelerometerdata, test av prestanda för den trådlösa uppkopplingen och ett försök att följa givarens position på bandet.

6.1 Datainsamling

Innan tester utförs på conveyorbandet så testades utrustningen tillsammans med en PC för att säkerställa att kommunikationen fungerade som avsett. Testet syftar även till att säkerställa att sensorn uppdateras med den uppdateringsfrekvens som är inställd i konfigureringsmenyn, samt att dataöverföringen mellan enheten och servern sker med samma frekvens.

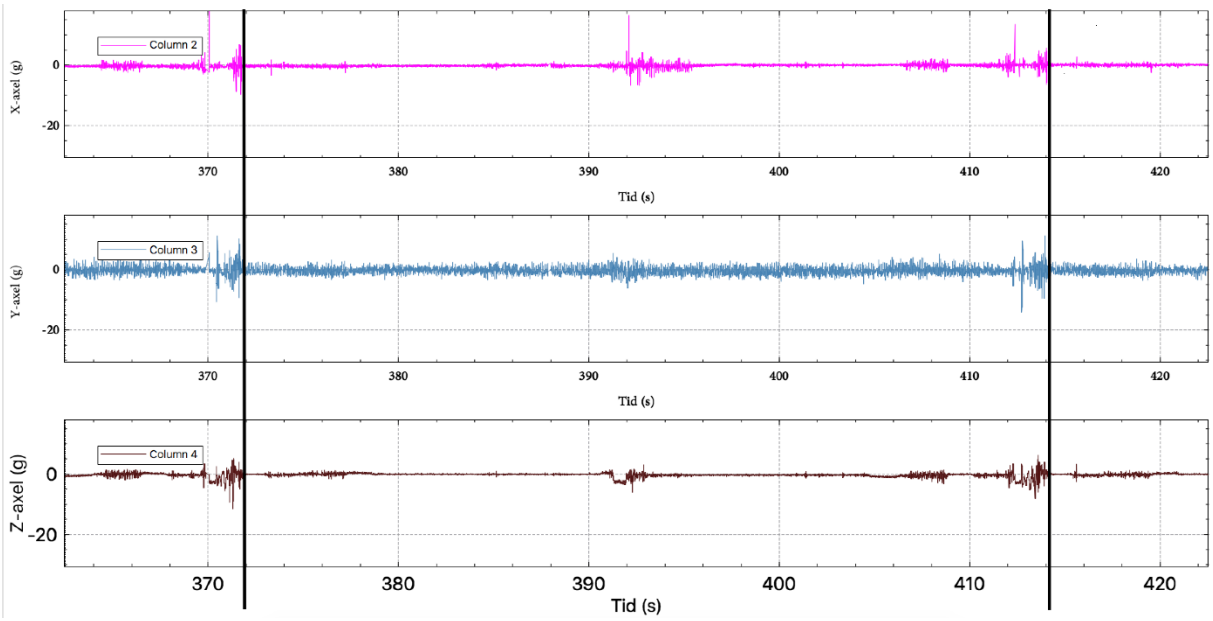
Första testet gjordes utan nätverksuppkoppling där inkommande mätdata visades i den seriella monitorn via USB kommunikation. För att kunna säkerställa att rätt överföringsfrekvens erhålls, samt att kunna presentera de erhållna mätpunkterna med en referens, så skickas en tidsvektor med i strängen. Med tidsvektorn integrerad så kan mängden datapunkter per sekund säkerställas och därmed frekvensen.



Figur 25: Mätdata i den seriella monitorn. De olika datapunkterna skiljs av ett semikolon.

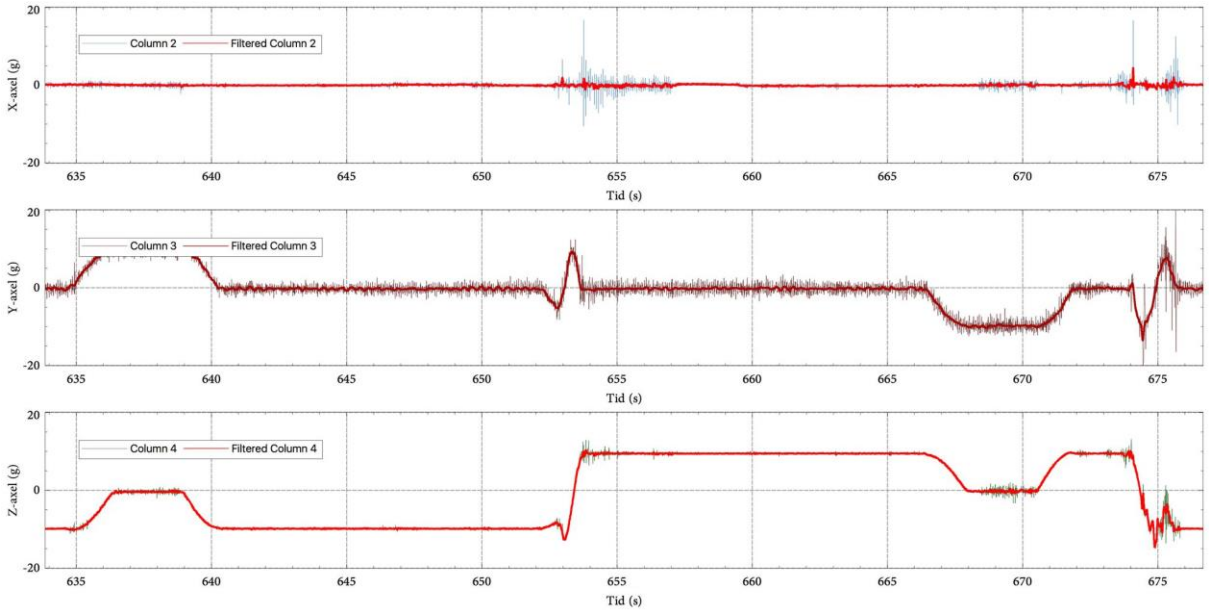
Efter säkerställd kommunikation fortgick testerna på ett transportband i labbet. På samma sätt men med en wifi uppkoppling och dataöverföring via TCP-protokoll. Uppkopplingen visar inga felaktigheter utan fungerar som planerat.

För att kunna avläsa mätdata på en PC används analysverktyget kst. I kst kan mätdata visualiseras i realtid men även i efterhand med insamlade datapunkter.



Figur 26: Mätpunkter från ett test på bandet med inställningen linjär acceleration. Mätpunkterna mellan de två svarta avgränsningarna representerar ett varv.

Programvaran kan även användas för att filtrera mätdata. Tester med implementering av lågpassfilter på de överförda datapunkterna har genomförts i ett försök att försöka få bort de mest högfrekventa störningarna. I figuren nedan visas hur datapunkterna har filtrerats med ett lågpassfilter som har en cutoff-frequency på 20Hz och en filterorder på fyra. Även de ofiltrerade datapunkterna finns med för att visa på hur mätdata förändras.



Figur 27: Den tjocka linjen representerar filtrerade mätdata och den tunna representerar ofiltrerad mätdata.

Tester har även utförts med företagets databas som server och data har då analyserats i analysverktyget Grafana där en graf kan plottas i webbläsaren.

Vidare gjordes tester på att nå högsta möjliga uppdateringsfrekvens inkluderat både datainsamling och dataöverföring till server. Målet för datainsamling är att uppnå en uppdateringsfrekvens på 1000Hz. På grund av den ”clock stretching” som krävs i I2C kommunikationen så uppstår problem att uppnå maximal hastighet.

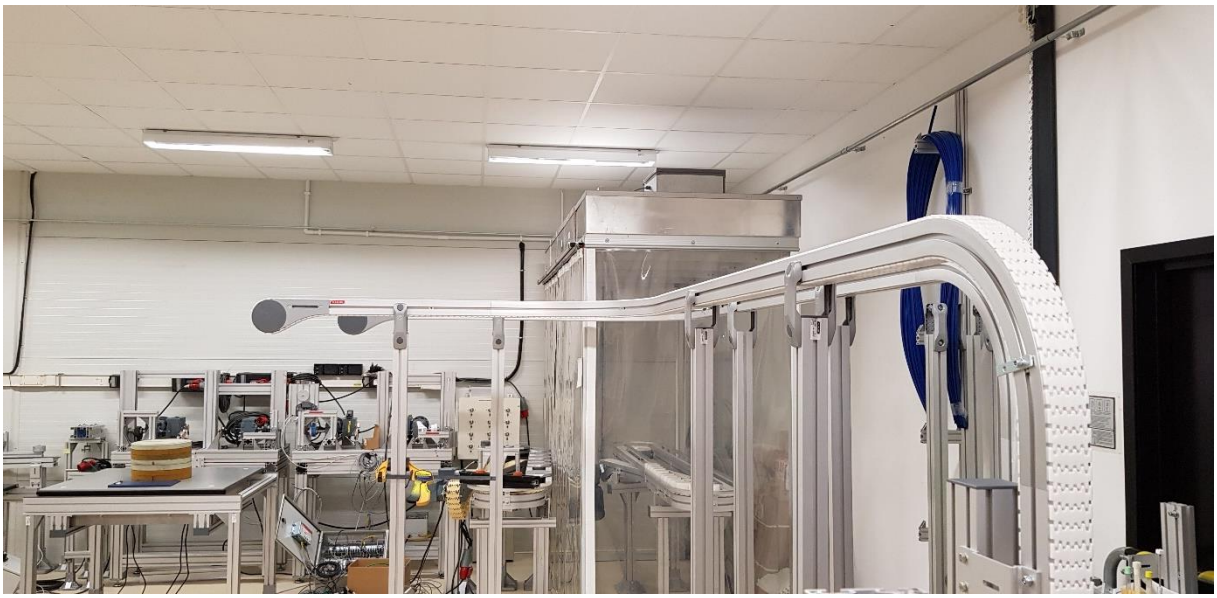
6.2 Tracking

I tidigare experiment har det fastställts att man kan se när givaren befinner sig i en kurva då datapunkterna från gyroskopet avviker. Ett annat intressant sätt att ta reda på var givaren befinner sig är att genom att använda sig av en beräkningsmetod som kallas ”attitude and heading reference system”, även kallat AHRS. Algoritmen använder sig vanligtvis av datapunkter från accelerometer och från gyroskop för att beräkna position och nuvarande riktning på det objekt man är intresserad av.

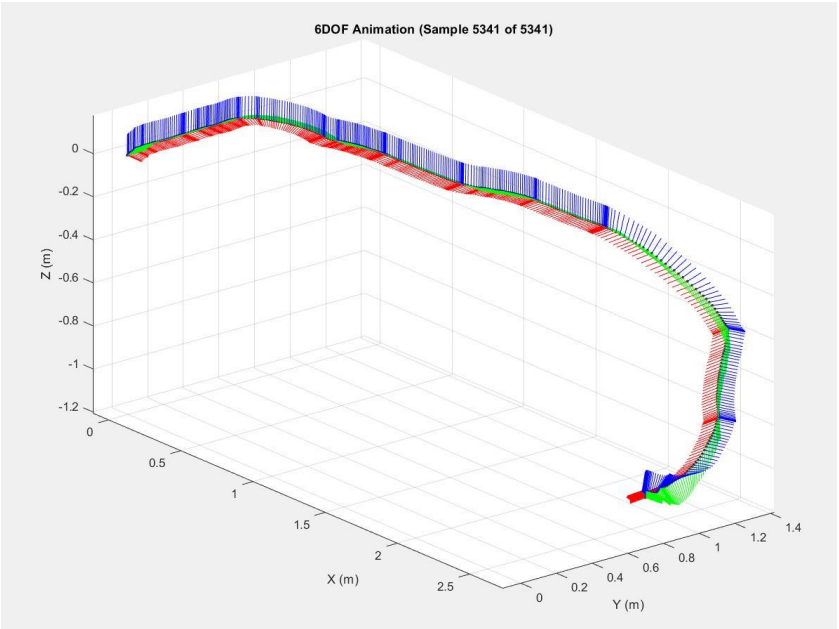
I det här projektet har en open source algoritm används för att experimentera med tracking [15]. Algoritmen som används är från ett projekt som genomfördes av xioTechnologies i september 2013. [16]. Programkoden som finns tillgänglig, som använder algoritmen, används för att följa en person som går i ett rum. Då många har misslyckats med att skapa en bra algoritm så lyckas denna för att den arbetar inom en viss periodtid som hjälper till att integrera bort det fel som ofta uppstår vid kontinuerlig användning. Det vill säga att rörelsen sker med en liten paus som gör det möjligt att integrera bort felen.

Det initiala testet gjordes utan någon paus i rörelsen för att testa möjligheten att spåra givaren kontinuerligt. Detta test gav ingen vettig information. Algoritmen klarade av att spåra enheten drygt en meter innan de integrerade felen vart alldeles för stora. Nästa test gjordes därför med planerade stopp så att algoritmen har tid att kompensera för de fel som tillkommer vid kontinuerlig drift.

Detta test fungerar bra när givaren befinner sig i positivt läge, det vill säga då givaren är på ovansidan av bandet. När man har kört testet och för in datapunkterna i matlab och kör algoritmen kan man se att positionen stämmer relativt bra överens med verkligheten, det finns en viss avvikelse i sträckan men formen är korrekt. Modellen visar att den sista raksträckan ska vara 1,4 meter, men den är två meter i verkligheten. För att spåra givaren på ett bättre sätt krävs något typ av globalt eller lokalt positioneringssystem som kan samköras med algoritmen för att få ett bättre resultat. Den metod som borde vara bäst för ändamålet vore att använda sig av en ”prediction algoritm” där man kan använda sig av en 3D modell av banan för att förbättra resultatet.



Figur 28: Bild på det band där tester utförs.

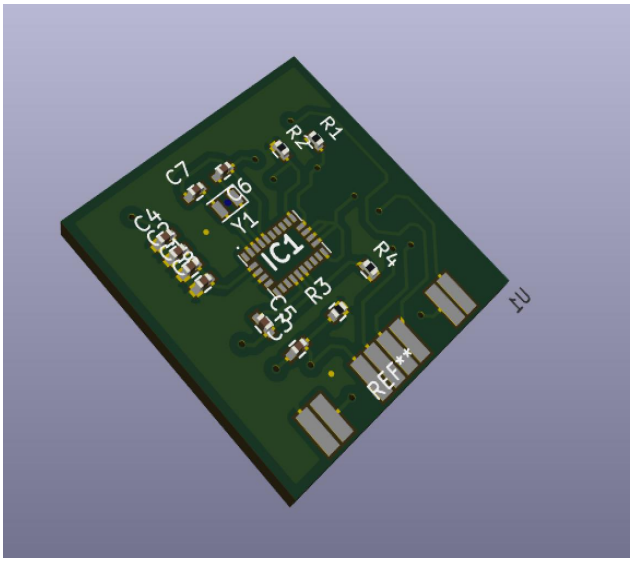


Figur 29: Animerad spårning av givaren med AHRS-algoritm.

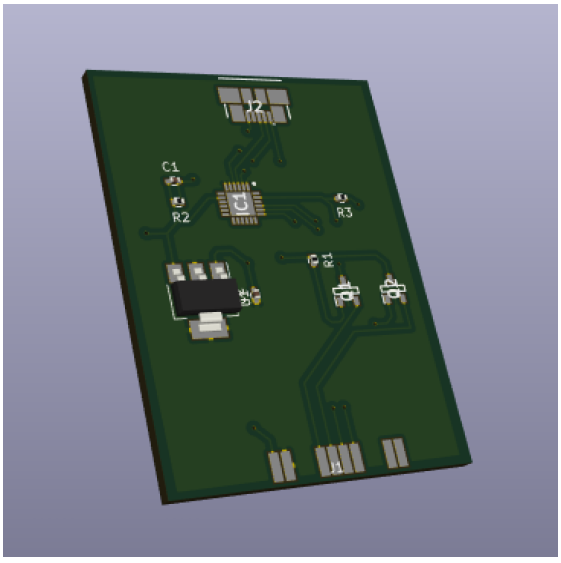
7 Produktprototyp

Efter att tester gjorts med testprototypen påbörjas arbetet med att göra ett förslag på hur en färdig prototyp skulle kunna se ut. Arbetet börjar med att analysera en standardlänk för att ta reda på hur stort utrymme som finns för elektroniken. Utrymmet som finns tillgängligt har en bestämd form, men efter några mätningar kan man konstatera att elektroniken måste få plats på ett kretskort med storleken 19,3x3,0x22,5 millimeter.

För att få plats med alla komponenter på den yta som är specificerad används bara de komponenter som är nödvändiga för att systemet ska fungera som det gör i testprototypen. Komponenter som USB-uttag, batterikontakt och tillhörande komponenter som krävs för dataöverföring från USB. Dessa komponenter placeras på ett separat kretskort som placeras i en dockningsstation som används för att föra över program från en dator samt att göra inställningar av enheten.

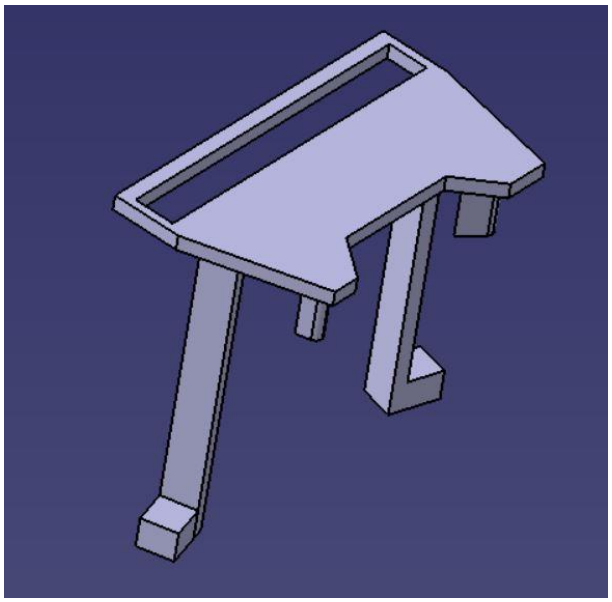


Figur 30: 3D-modell av huvudkortet.



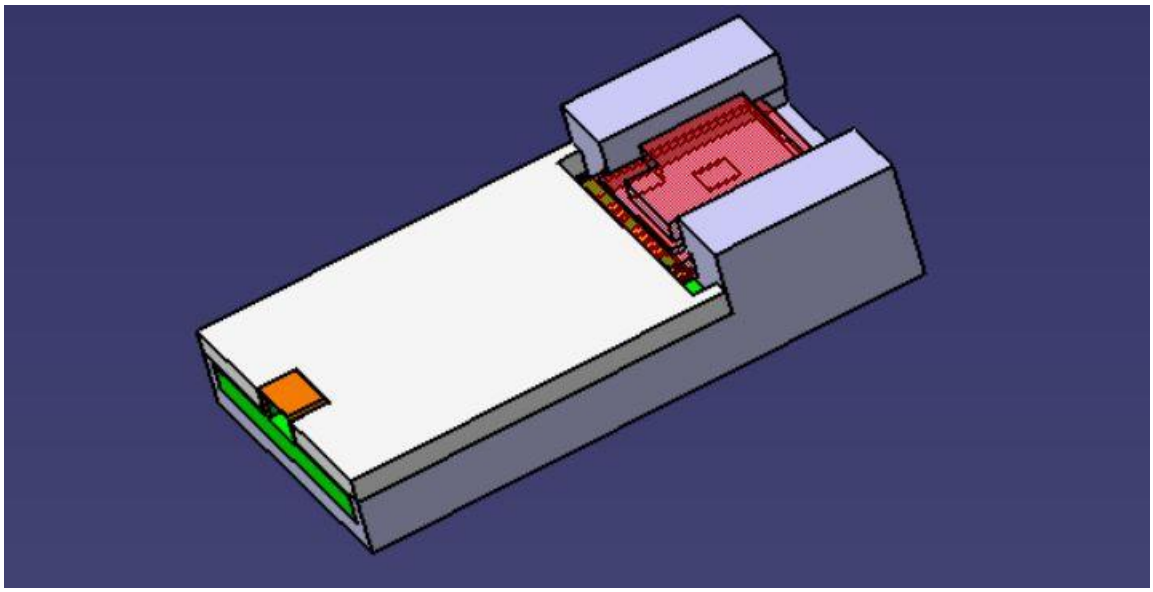
Figur 31: 3D-modell av dockningskortet.

När alla komponenter är på plats på huvudkortet kan man konstatera att det är något bredare än vad utrymmet tillåter. Därför måste länken konfigureras så att kortet får plats i länken. En millimeter av länken tas då bort från varje sida av det utrymme som finns. För att fästa kortet designas ett lock med hakar som kan fästas i två hål som finns på undersidan av länken. Det skapas ett spår i locket där kortet kan limmas fast så att kortet sitter fast permanent i locket. Till sist skapas två styrande axlar för att hålla kortet på plats i planet. För att ta bort kortet från länken lossas hakarna och kortet lyfts ut. Cad modeller och färdigställd design och utformning utav kretskort finns framtaget och är monterade i en virtuell miljö.



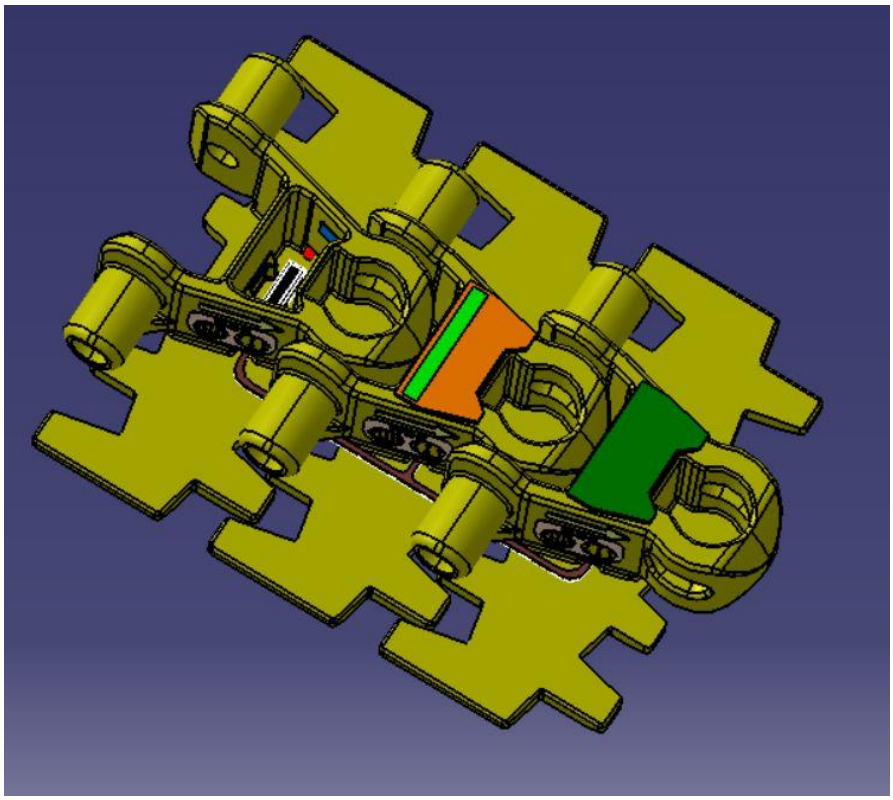
Figur 32: Lock som används för infästning av huvudkort.

För att ha möjlighet att föra över nya versioner av program men även för att konfigurera enheten har en dockningsstation designats. Basen som utgör dockningsstationen har två spår i olika nivåer. Den nedre nivån utgör ett utrymme där dockingskortet kan fästas och den övre nivån utgör ett utrymme där huvudenheten kan skjutas in. Locket som är placerat ovanför dockningskortet limmas fast för att skydda kortet från yttre påverkan. När huvudkortet skjut in i utrymmet får det kontakt med de stift som finns placerade på dockingskortet. Från stiften går signaler genom kretskortet till USB-kontakten som kommunicerar med datorn som är ansluten. Från datorn kan enhetens uppkoppling och noggrannhet konfigureras.



Figur 33: Figuren visar utseendet av dockningsstationen. Röd detalj är huvudkortet. Grön detalj är dockningskortet. De resterande detaljerna är själva dockningsstationen.

Konceptet består i standardutförande av två länkar i serie, huvudlänken som innehåller kretsen och en länk som innehåller ett batteri för strömförsörjning. För att fästa batteriet används ett liknande lock som används till huvudkortet. Till skillnad från det finns inget spår där kortet limmas utan batteriet limmas direkt på locket och fästs sedan på samma sätt. Ett koncept för en länk till finns tillgängligt, den länken innehåller en krets med trådtöjningsgivare för att kunna mäta deformationen av länken vid drift. För att koppla samman dessa länkar används en databuss. Den består av fyra kablar, en för SDA, en för SCL, en för jord och en för drivspänning. De kablar som tillhör I2C överföringen går parallellt på ena sidan av länkens bas och de andra går parallellt på andra sidan.



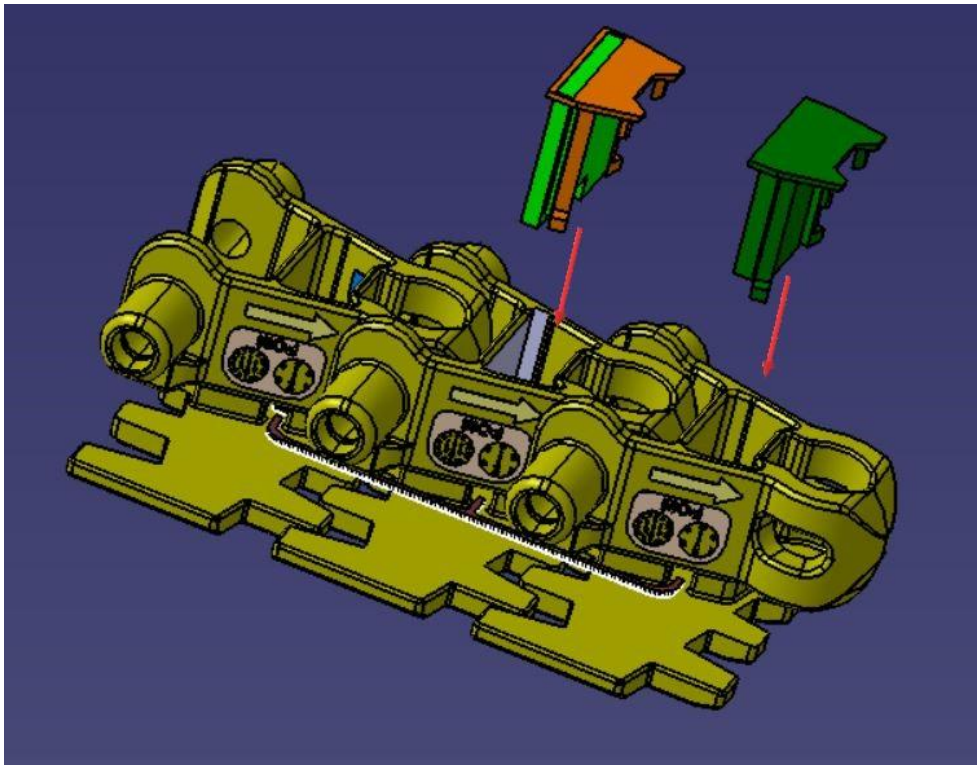
Figur 34: De tre olika länkarna sammansatta, vy från undersida. Första länken med grönt lock innehåller ett batteri. Den andra med orange lock innehåller huvudkortet som i figuren är ljust grön. Sista länken innehåller töjningsgivare.

8 Resultat

Vid projektets slut, finns en färdig testenhet och en virtuell produktprototyp.

Testenheten kan mäta acceleration och vinkelacceleration i alla reella axlar, och skicka det avlästa värdet till en server via wifi och TCP/IP-kommunikation. Som mottagande server kan en PC eller företagets databastjänster användas. Högsta uppmätta uppdateringsfrekvens mellan sensorn och huvudenheten är 400Hz. Frekvensanalys utav bandet visar dock att intressanta frekvenser finns under 30Hz.Fästen och hållare till huvudenheten och sensorn har tillverkats i plast genom 3D-printing. Indikatorlampor har anslutits för lättare kunna identifiera att enheten sänder mätdata och att sensorn fungerar korrekt. Enheten är testad i labbmiljö där den med god effekt sänt mätdata och visat sig ge företaget större insyn i bandets dynamik.

Den virtuella produktprototypen har utvecklats utifrån samma komponenter som i testenheten, men med bortskalade funktioner och komponenter som inte är nödvändiga för ändamålet. En kretskortdesign har tagits fram där både sensorn (BNO 055) och microcontrollern (ESP32) sitter på samma kretskort och som har en storlek (21,75mmx3,0mmx21,5mm) anpassad till att passa i en av företagets länkar (x85). USB-anslutning finns inte direkt på kortet utan istället har en dockningsstation tagits fram för kretskortet för att möjliggöra överföring av program samt konfigurering. Hållare och fästen för kretskorten är framtagna som 3D CAD-modeller och kan tillverkas i plast med 3D-printer.



Figur 35: Figuren visar hur batteriet och enheten monteras i länkarna. Enhet till vänster och batteri till höger.

9 Slutsats/Diskussion

Sensorn *BNO 055* har enligt sitt eget produktblad en kapacitet på upp till 1000*Hz*, men då sensorn kräver ”*clock-stretching*” vid I2C-kommunikationen så skapas en begränsning och max genomsnittliga uppdateringsfrekvens blir då 400*Hz*. Skulle högre uppdateringsfrekvens vilja uppnås så krävs en annan sensor utan inbyggd microcontroller.

Valet att använda sig utav utvecklingsplattformen ESP-IDF istället för Arduino innebar det tog lång tid att lära sig plattformen som i sig är mer komplex än Arduino. Fördelen med ESP-IDF är att större valmöjligheter finns i att utveckla ett mer skräddarsytt program för enheten och att vidare utveckling blir lättare. Arduino skulle kunna ha använts i ett initialt skede för att testa sensorer för att snabbare bilda sig en uppfattning.

Trots försök att minimera storleken på kretskortet för prototypen, så krävs en mindre förändring på länken x85 med 1mm för att den skall passa. Målet var att designa en modul som skulle kunna installeras i en länk utan att behöva modifiera länken. För att nå det resultat så krävs mer tid till att i så fall designa sin egen esp32-modul istället för att köpa en färdigbyggd, det skulle också skapa förutsättningar till att implementera på mindre länk-modeller.

9.1 Vidareutveckling

Microcontrollern har möjlighet till att ta emot och skicka analoga signaler, vilket öppnar upp för möjligheten att ansluta en töjningsgivare. En töjningsgivare skulle kunna ge mätdata som möjliggör beräkningar av belastning utav band och länk. Vidare skulle även en hallsensor kunna implementeras för att identifiera slitage, slitage på länkar men även på glidlist.

Att realisera prototypen skulle även kunna vara nyttigt för mer än det tänkta ändamålet. Den skulle kunna ersätta testenheten och användas på samma sätt, monterad ovanpå länken. Den skulle göra det möjligt att ha en utanpåliggande sensorenheten vid tillfällen då utrymmet kring bandet är begränsat.

Ett alternativ till att försöka anpassa sensorenheten till att passa i en befintlig länk vore att istället anpassa länken efter kretskortet. En special-länk skulle kunna utformas så att elektroniken lättare kan byggas samman med länken. Dock skulle det kräva att beräkningar görs för den nya länken så att den håller samma hållfasthet som övriga länkar, så den inte blir den svaga länken.

En vidareutveckling av hantering utav mätdata och konfigurering av enheten, vore att möjliggöra trådlös kommunikation med en smartphone. En applikation där du kan ställa in önskade inställningar och också läsa de insamlade mätvärdena. ESP32-modulen kan kommunicera över Bluetooth vilket skulle kunna passa för ändamålet, då man kan kommunicera med enheten även om den är uppkopplad via wifi med en server.

Försöken av tracking gav inte tillräckligt bra utslag. För att kunna skapa en mer precis och korrekt bana så skulle nog ett kompleterande UVB-trackingsystem tillsammans med AHRS algoritmen kunna ge en mer precis bild. Även en ”prediction” (en 3D-modell av banan) skulle kunna vara ett verktyg för att korrelera kordinaterna för att identifiera position.

Referenser

[1] Kolban's Book on ESP32, Neil Kolban, Texas, United States of America, 2018. [Online] Available: https://leanpub.com/kolban-ESP32

[2] Adafruit Industries, "Adafruit HUZZAH32 – ESP32 Feather Board",2017. [Digital Image]. Available: https://www.adafruit.com/product/3405 Downloaded: 2018-05-14.

[3] Bosch Sensortec, "BNO055 Intelligent 9-axis absolute orientation sensor", BNO055 datasheet, Nov 2014. [Online].
Available:
https://aebst.resource.bosch.com/media/_tech/media/datasheets/BST_BNO055_DS000_14.pdf

[4] Adafruit Industries, "Adafruit 9-DOF Absolute Orientation IMU Fusion Breakout – BNO055", 2015. [Elektronisk bild]. Available: https://www.adafruit.com/product/2472 Downloaded: 2018-05-14.

[5] Eclipse Cpp Oxygen, Version Oxygen.2 Release 4.7.2, [Software], Available: https://www.eclipse.org/downloads/

[6] Espressif Systems. (2018). esp-idf. [Online]. Available: https://github.com/espressif/esp-idf Downloaded: 2018-03-27.

[7] freeRTOS, Version 10.0.1, [Software], Available: https://www.freertos.org

[8] Kst, Version 2.0.x, [Software], Berlin, Germany: K Desktop Environment incorporated society, 20xx.

[9] Wire shark, Version 2.6.0, [Software], Available: https://www.wireshark.org

[10] MINGW32, Version 4.8.1, [Software], Available: http://www.mingw.org

[11] Grafana, Version 5.1.3, [Software], Available: https://grafana.com/

[12] RabbitMQ, Version 3.7.4, [Software], Available: https://www.rabbitmq.com/

[13] VibSensor, Version 2.0.0, [Software], Available: http://www.now-instruments.com/products/3-vibsensor

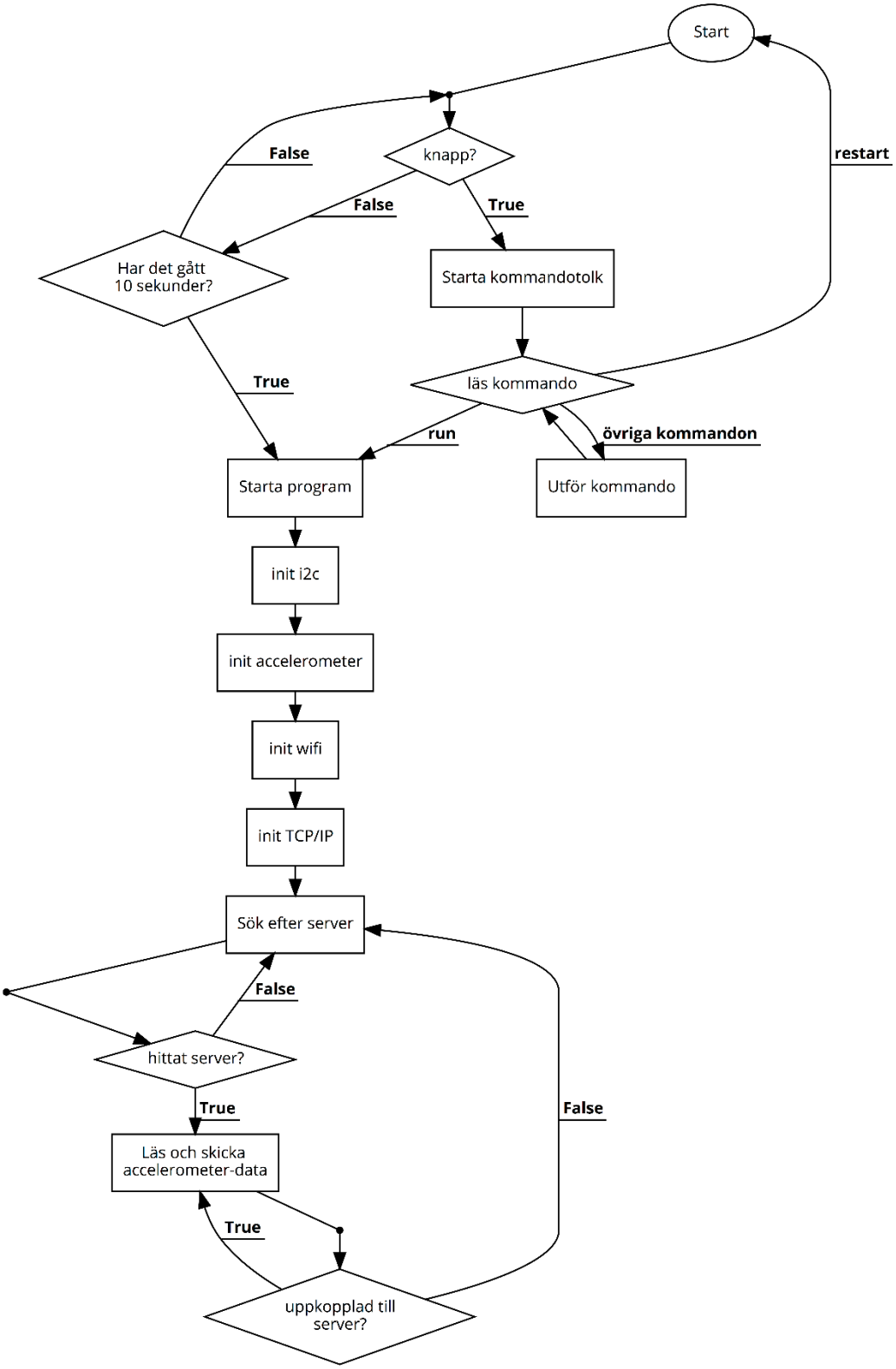
[14] "UDP och TCP – två sätt att skicka trafik", Fixa nätet, May. 18, 2018. [Online] Available: https://www.fixanatet.se/portar-och-nat/udp-och-tcp-tva-satt-att-skicka-trafik/

[15] xioTechnologies. (2011). Gait-Tracking-With-x-IMU. [Online]. Available: https://github.com/xioTechnologies/Gait-Tracking-With-x-IMU/tree/master/Gait%20Tracking%20With%20x-IMU Downloaded: 2018-03-19

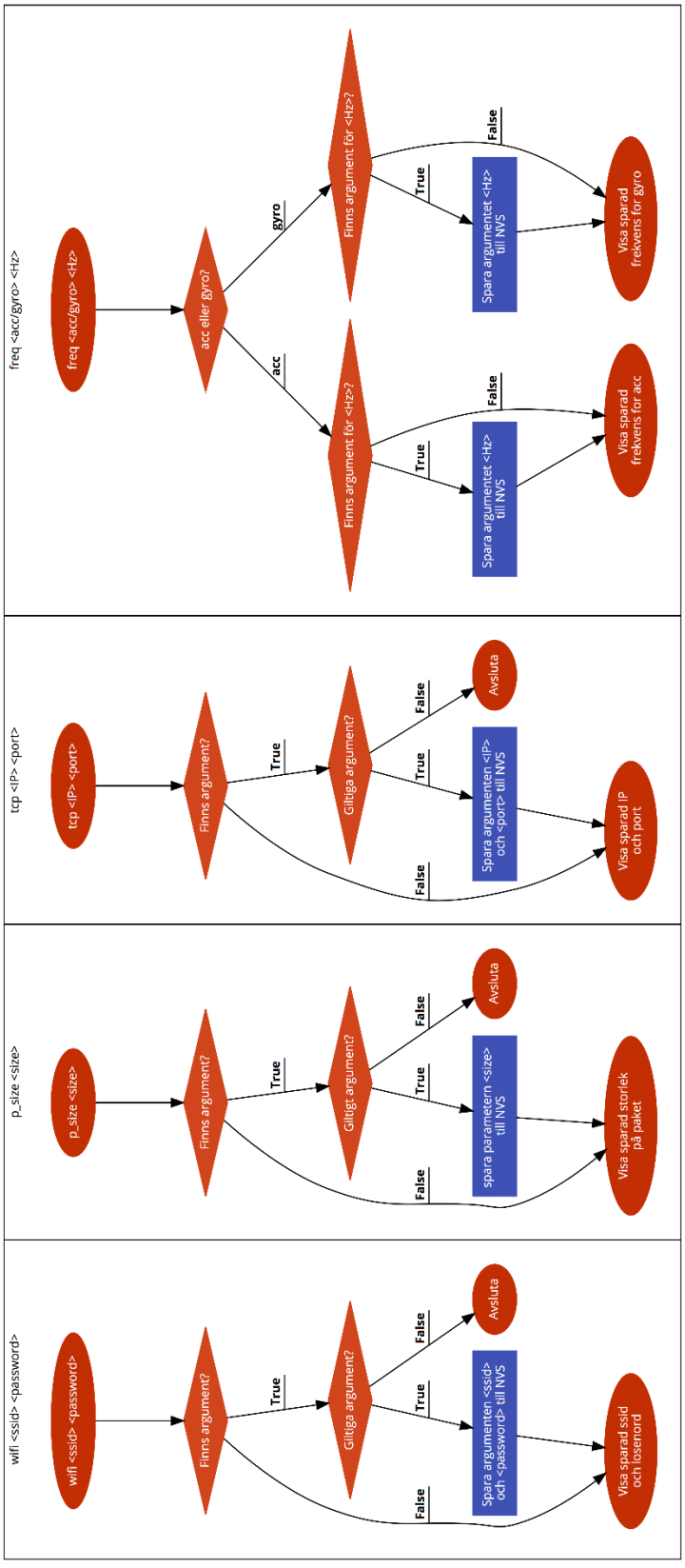
[16] Sebastian O.H Madgwick, "An efficient orientation filter for inertial and inertial/magnetic sensor arrays" M.S thesis, X-IO Technologies, United Kingdom, 2010.[Online]. Available: http://x-io.co.uk/open-source-imu-and-ahrs-algorithms/

Bilaga A: Flödeschema för program

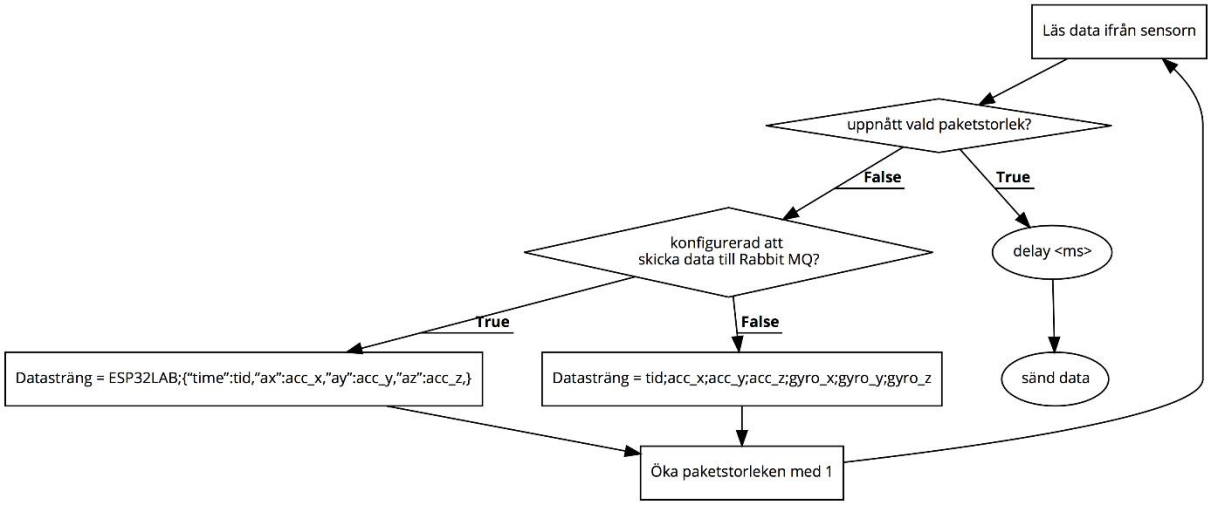
Huvudprogram:



Exempel på kommandon:

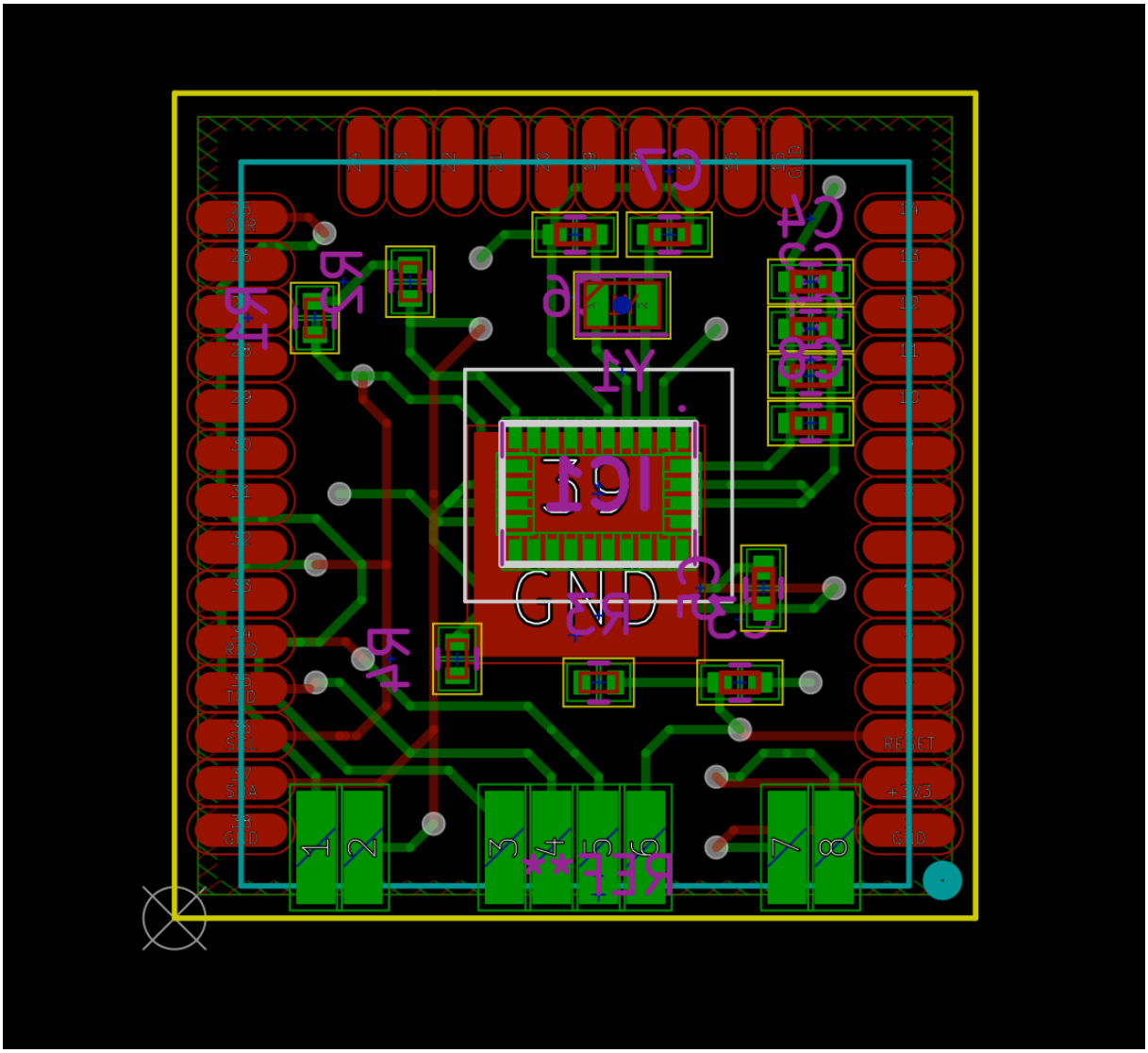


Läsa/skicka från sensorn:

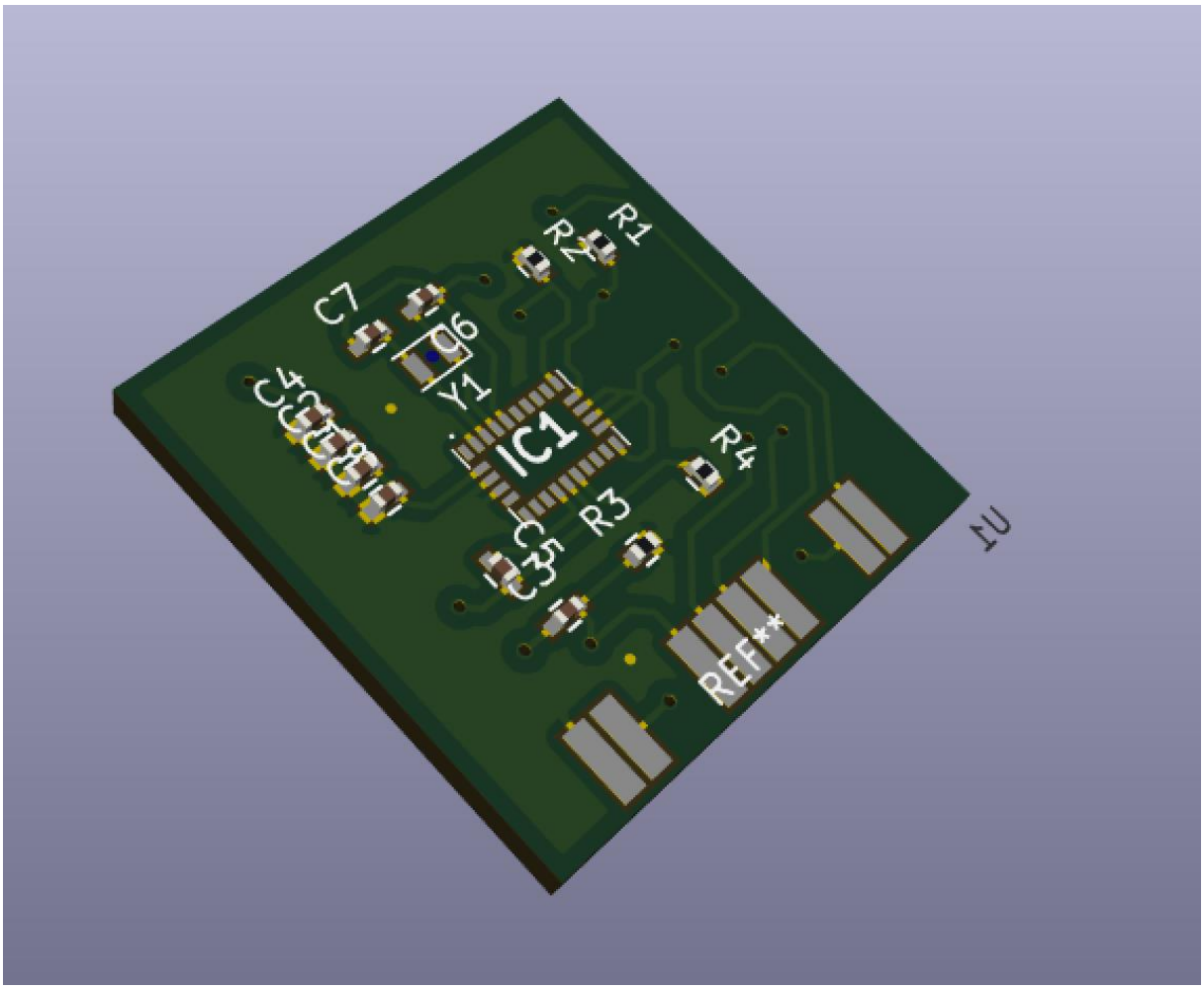


Bilaga B: Kretskort

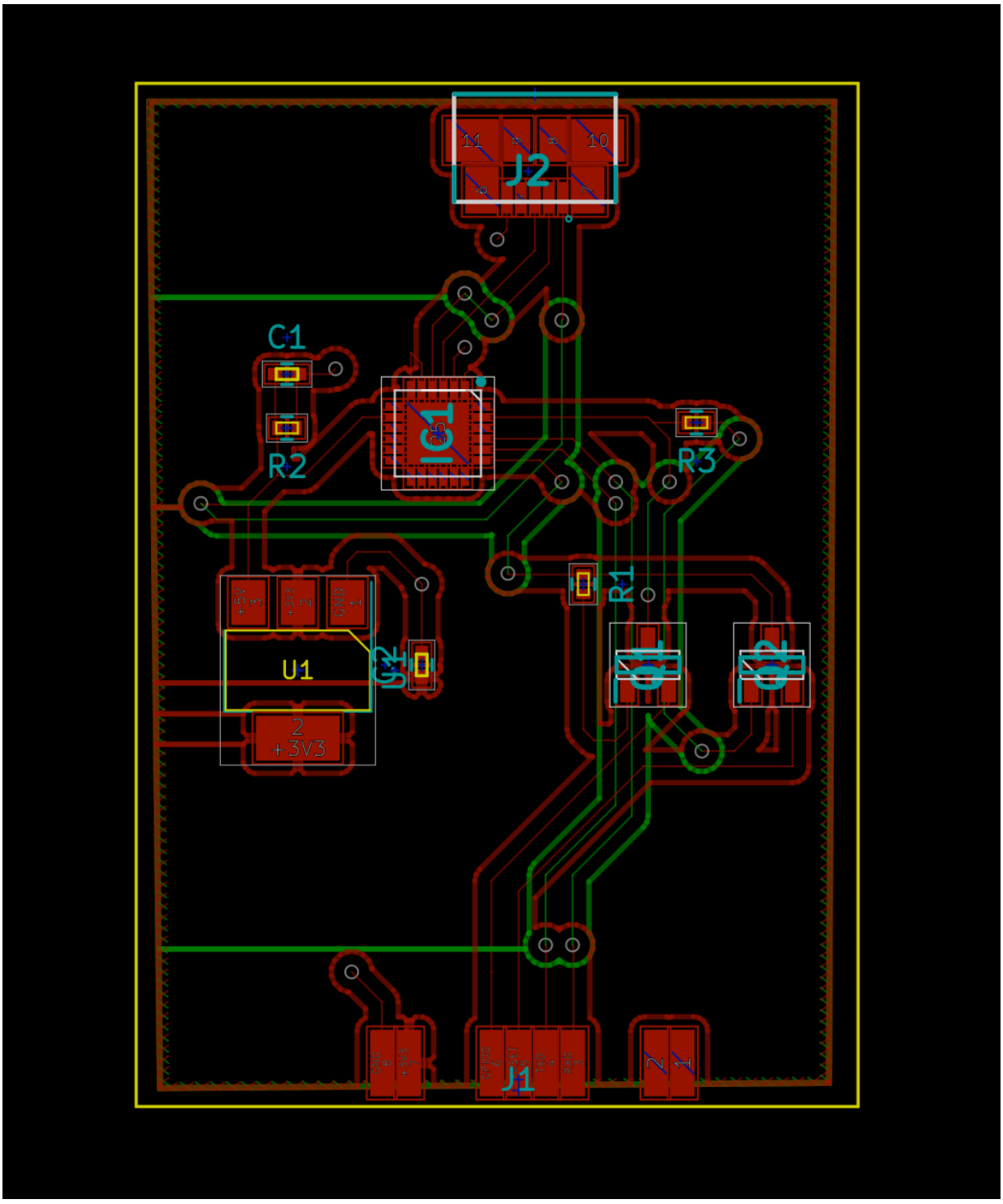
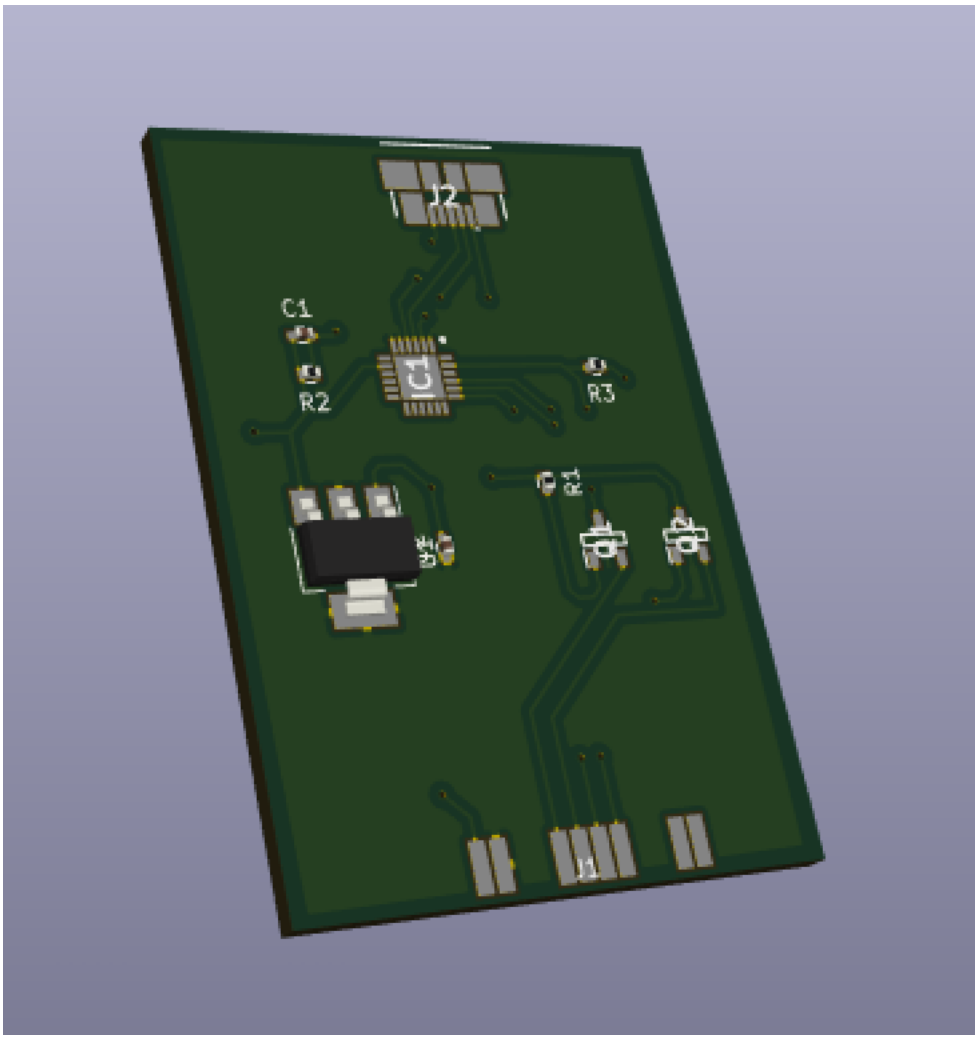
Kretsschema för SmartLink:



PCB-kort för SmartLink:



Kretsschema för dockiningsstation:



PCB-kort för dockningsstationen: